# Exhibit B



# (12) United States Patent Katzer

(10) Patent No.:

US 6,530,329 B2

(45) Date of Patent:

\*Mar. 11, 2003

### (54) MODEL TRAIN CONTROL SYSTEM

(75) Inventor: Matthew A. Katzer, 1416 NW.
Benfield Dr., Portland, OR (US) 97229

(73) Assignee: Matthew A. Katzer, Hillsboro, OR

(US)

(\*) Notice: Subject to any disclaimer, the term of this

patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

This patent is subject to a terminal disclaimer.

(21) Appl. No.: 10/124,878

(22) Filed: Apr. 17, 2002

(65) Prior Publication Data

US 2002/0170458 A1 Nov. 21, 2002

### Related U.S. Application Data

(63)	Continuation of application No. 09/858,222, filed on Apr.
` '	17. 2002, now Pat. No. 6,460,467.

(51)	Int. Cl.7		A63H	19/00
------	-----------	--	------	-------

(52) **U.S. Cl.** ...... **105/1.5**; 246/167 R; 246/197; 246/62

### (56) References Cited

### U.S. PATENT DOCUMENTS

3,944,980 A	3/19/0	Staptes
3,976,272 A	8/1976	Murray et al.
4,307,302 A	12/1981	Russell
4,853,883 A	* 8/1989	Nickles et al 348/121
5,072,900 A	12/1991	Malon

5,475,818	Α		12/1995	Molyneaux et al 701/20
5,493,642	Α		2/1996	Dunsmuir et al.
5,638,522	Α		6/1997	Dunsmuir et al.
5,681,015	Α	*	10/1997	Kull 246/167 R
5,696,689	Α		12/1997	Okumura et al.
5,787,371	Α	*	7/1998	Balukin et al 246/187 A
5,828,979	Α		10/1998	Ploivka et al.
5,896,017	Α		4/1999	Severson et al.
5,940,005	Α		8/1999	Severson et al.
5,952,797	Α		9/1999	Rossler
6,065,406	Α	*	5/2000	Katzer 105/1.4
6,267,061	<b>B</b> 1		7/2001	Katzer
6,270,040	<b>B</b> 1		8/2001	Katzer

#### OTHER PUBLICATIONS

Chapell, David, Understanding ActiveX and OLE, 1996, Microsoft Press, Redmond.

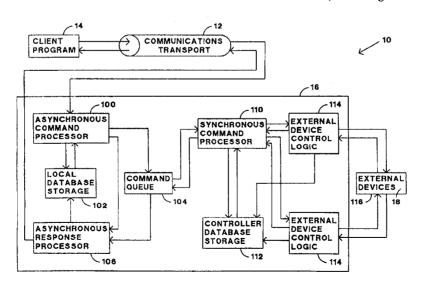
\* cited by examiner

Primary Examiner—William A. Cuchlinski, Jr.
Assistant Examiner—Olga Hernandez
(74) Attorney, Agent, or Firm—Chernoff, Vilhauer,
McClung & Stenzel, LLP

#### (57) ABSTRACT

A system which operates a digitally controlled model rail-road transmitting a first command from a first client program to a resident external controlling interface through a first communications transport. A second command is transmitted from a second client program to the resident external controlling interface through a second communications transport. The first command and the second command are received by the resident external controlling interface which queues the first and second commands. The resident external controlling interface sends third and fourth commands representative of the first and second commands, respectively, to a digital command station for execution on the digitally controlled model railroad.

### 27 Claims, 3 Drawing Sheets

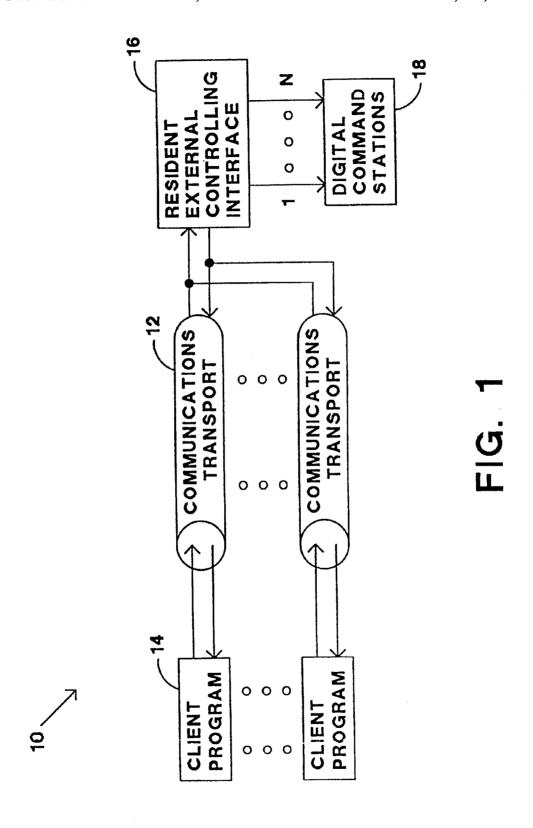


U.S. Patent

Mar. 11, 2003

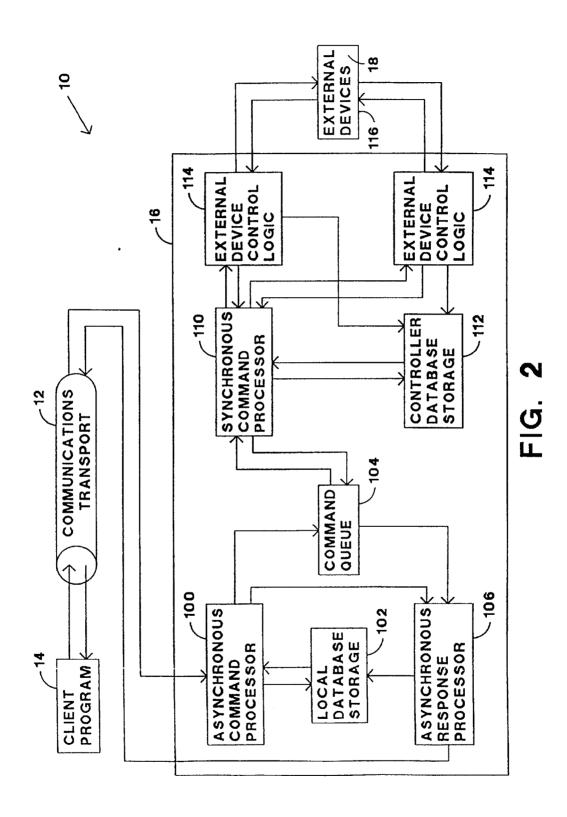
Sheet 1 of 3

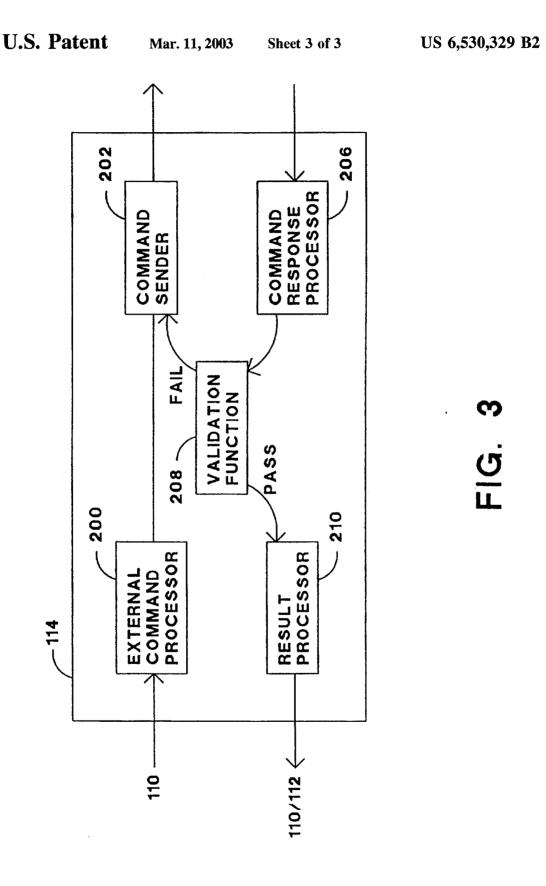
US 6,530,329 B2



**U.S. Patent** Mar. 11, 2003 Sheet 2

Sheet 2 of 3 US 6,530,329 B2





#### 1

### MODEL TRAIN CONTROL SYSTEM

This application is a continuation of U.S. patent application Ser. No. 09/858,222 filed on Apr. 17, 2002 U.S. Pat. No. 6.460.467.

#### BACKGROUND OF THE INVENTION

The present invention relates to a system for controlling a model railroad.

Model railroads have traditionally been constructed with of a set of interconnected sections of train track, electric switches between different sections of the train track, and other electrically operated devices, such as train engines and draw bridges. Train engines receive their power to travel on the train track by electricity provided by a controller through the track itself. The speed and direction of the train engine is controlled by the level and polarity, respectively, of the electrical power supplied to the train track. The operator manually pushes buttons or pulls levers to cause the 20 switches or other electrically operated devices to function, as desired. Such model railroad sets are suitable for a single operator, but unfortunately they lack the capability of adequately controlling multiple trains independently. In addition, such model railroad sets are not suitable for being 25 controlled by multiple operators, especially if the operators are located at different locations distant from the model railroad, such as different cities.

A digital command control (DDC) system has been developed to provide additional controllability of individual train 30 engines and other electrical devices. Each device the operator desires to control, such as a train engine, includes an individually addressable digital decoder. A digital command station (DCS) is electrically connected to the train track to provide a command in the form of a set of encoded digital bits to a particular device that includes a digital decoder. The digital command station is typically controlled by a personal computer. A suitable standard for the digital command control system is the NMRA DCC Standards, issued March 1997, and is incorporated herein by reference. While providing the ability to individually control different devices of the railroad set, the DCC system still fails to provide the capability for multiple operators to control the railroad devices, especially if the operators are remotely located from the railroad set and each other.

DigiToys Systems of Lawrenceville, Ga. has developed a software program for controlling a model railroad set from a remote location. The software includes an interface which allows the operator to select desired changes to devices of the railroad set that include a digital decoder, such as increasing the speed of a train or switching a switch. The software issues a command locally or through a network, such as the internet, to a digital command station at the railroad set which executes the command. The protocol used by the software is based on Cobra from Open Management 55 Group where the software issues a command to a communication interface and awaits confirmation that the command was executed by the digital command station. When the software receives confirmation that the command executed, the software program sends the next command through the communication interface to the digital command station. In other words, the technique used by the software to control the model railroad is analogous to an inexpensive printer where commands are sequentially issued to the printer after the previous command has been executed. Unfortunately, it has been observed that the response of the model railroad to the operator appears slow, especially over a distributed

#### 2

network such as the internet. One technique to decrease the response time is to use high-speed network connections but unfortunately such connections are expensive.

What is desired, therefore, is a system for controlling a model railroad that effectively provides a high-speed connection without the additional expense associated therewith.

The foregoing and other objectives, features, and advantages of the invention will be more readily understood upon consideration of the following detailed description of the invention, taken in conjunction with the accompanying drawings.

#### SUMMARY OF THE PRESENT INVENTION

The present invention overcomes the aforementioned drawbacks of the prior art, in a first aspect, by providing a system for operating a digitally controlled model railroad, that includes transmitting a first command from a first client program to a resident external controlling interface through a first communications transport. A second command is transmitted from a second client program to the resident external controlling interface through a second communications transport. The first command and the second command are received by the resident external controlling interface which queues the first and second commands. The resident external controlling interface sends third and fourth commands representative of the first and second commands, respectively, to a digital command station for execution on the digitally controller model railroad.

Incorporating a communications transport between the multiple client program and the resident external controlling interface permits multiple operators of the model railroad at locations distant from the physical model railroad and each other. In the environment of a model railroad club where the members want to simultaneously control devices of the same model railroad layout, which preferably includes multiple trains operating thereon, the operators each provide commands to the resistant external controlling interface, and hence the model railroad In addition by queuing by commands at a single resident external controlling interface permits controlled execution of the commands by the digitally controlled model railroad, would may otherwise conflict with one another.

In another aspect of the present invention the first command is selectively processed and sent to one of a plurality
of digital command stations for execution on the digitally
controlled model railroad based upon information contained
therein. Preferably, the second command is also selectively
processed and sent to one of the plurality of digital command
sations for execution on the digitally controlled model
railroad based upon information contained therein. The
resident external controlling interface also preferably
includes a command queue to maintain the order of the
commands.

The command queue also allows the sharing of multiple devices, multiple clients to communicate with the same device (locally or remote) in a controlled manner, and multiple clients to communicate with different devices. In other words, the command queue permits the proper execution in the cases of: (1) one client to many devices, (2) many clients to one device, and (3) many clients to many devices.

In yet another aspect of the present invention the first command is transmitted from a first client program to a first processor through a first communications transport. The first command is received at the first processor. The first processor provides an acknowledgement to the first client program through the first communications transport indicating that

the first command has properly executed prior to execution of commands related to the first command by the digitally controlled model railroad. The communications transport is preferably a COM or DCOM interface.

The model railroad application involves the use of 5 extremely slow real-time interfaces between the digital command stations and the devices of the model railroad. In order to increase the apparent speed of execution to the client, other than using high-speed communication interfaces, the resident external controller interface receives 10 the command and provides an acknowledgement to the client program in a timely manner before the execution of the command by the digital command stations. Accordingly, the execution of commands provided by the resident external controlling interface to the digital command stations 15 occur in a synchronous manner, such as a first-in-first-out manner. The COM and DCOM communications transport between the client program and the resident external controlling interface is operated in an asynchronous manner, namely providing an acknowledgement thereby releasing 20 the communications transport to accept further communications prior to the actual execution of the command. The combination of the synchronous and the asynchronous data communication for the commands provides the benefit that the operator considers the commands to occur nearly instan- 25 taneously while permitting the resident external controlling interface to verify that the command is proper and cause the commands to execute in a controlled manner by the digital command stations, all without additional high-speed communication networks. Moreover, for traditional distributed 30 software execution there is no motivation to provide an acknowledgment prior to the execution of the command because the command executes quickly and most commands are sequential in nature. In other words, the execution of the next command is dependent upon proper execution of the 35 prior command so there would be no motivation to provide an acknowledgment prior to its actual execution.

# BRIEF DESCRIPTION OF THE SEVERAL VIEWS OF THE DRAWINGS

FIG. 1 is a block diagram of an exemplary embodiment of a model train control system.

FIG. 2 is a more detailed block diagram of the model train control system of FIG. 1 including external device control dogic.

FIG. 3 is a block diagram of the external device control logic of FIG. 2.

# DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

Referring to FIG. 1, a model train control system 10 includes a communications transport 12 interconnecting a client program 14 and a resident external controlling interface 16. The client program 14 executes on the model 55 railroad operator's computer and may include any suitable system to permit the operator to provide desired commands to the resident external controlling interface 16. For example, the client program 14 may include a graphical interface representative of the model railroad layout where 60 the operator issues commands to the model railroad by making changes to the graphical interface. The client program 14 also defines a set of Application Programming Interfaces (API's), described in detail later, which the operator accesses using the graphical interface or other programs 65 such as Visual Basic, C++, Java, or browser based applications. There may be multiple client programs interconnected

with the resident external controlling interface 16 so that multiple remote operators may simultaneously provide control commands to the model railroad.

The communications transport 12 provides an interface between the client program 14 and the resident external controlling interface 16. The communications transport 12 may be any suitable communications medium for the transmission of data, such as the internet, local area network, satellite links, or multiple processes operating on a single computer. The preferred interface to the communications transport 12 is a COM or DCOM interface, as developed for the Windows operating system available from Microsoft Corporation. The communications transport 12 also determines if the resident external controlling interface 16 is system resident or remotely located on an external system. The communications transport 12 may also use private or public communications protocol as a medium for communications. The client program 14 provides commands and the resident external controlling interface 16 responds to the communications transport 12 to exchange information. A description of COM (common object model) and DCOM (distributed common object model) is provided by Chappel in a book entitled Understanding ActiveX and OLE, Microsoft Press, and is incorporated by reference herein.

Incorporating a communications transport 12 between the client program(s) 14 and the resident external controlling interface 16 permits multiple operators of the model railroad at locations distant from the physical model railroad and each other. In the environment of a model railroad club where the members want to simultaneously control devices of the same model railroad layout, which preferably includes multiple trains operating thereon, the operators each provide commands to the resistant external controlling interface, and hence the model railroad.

The manner in which commands are executed for the model railroad under COM and DCOM may be as follows. The client program 14 makes requests in a synchronous manner using COM/DCOM to the resident external interface controller 16. The synchronous manner of the request is the technique used by COM and DCOM to execute commands. The communications transport 12 packages the command for the transport mechanism to the resident external controlling interface 16. The resident external controlling interface 16 then passes the command to the digital command stations 18 which in turn executes the command. After the digital command station 18 executes the command an acknowledgement is passed back to the resident external controlling interface 16 which in turn passes an acknowledgement to the client program 14. Upon receipt of the 50 acknowledgement by the client program 14, the communications transport 12 is again available to accept another command. The train control system 10, without more, permits execution of commands by the digital command stations 18 from multiple operators, but like the DigiToys Systems' software the execution of commands is slow.

The present inventor came to the realization that unlike traditional distributed systems where the commands passed through a communications transport are executed nearly instantaneously by the server and then an acknowledgement is returned to the client, the model railroad application involves the use of extremely slow real-time interfaces between the digital command stations and the devices of the model railroad. The present inventor came to the further realization that in order to increase the apparent speed of execution to the client, other than using high-speed communication interfaces, the resident external controller interface 16 should receive the command and provide an

4

acknowledgement to the client program 12 in a timely manner before the execution of the command by the digital command stations 18. Accordingly, the execution of commands provided by the resident external controlling interface 16 to the digital command stations 18 occur in a synchronous manner, such as a first-in-first-out manner. The COM and DCOM communications transport 12 between the client program 14 and the resident external controlling interface 16 is operated in an asynchronous manner, namely providing an acknowledgement thereby releasing the communications transport 12 to accept further communications prior to the actual execution of the command. The combination of the synchronous and the asynchronous data communication for the commands provides the benefit that the operator considers the commands to occur nearly instantaneously while permitting the resident external controlling interface 16 to verify that the command is proper and cause the commands to execute in a controlled manner by the digital command stations 18, all without additional highspeed communication networks. Moreover, for traditional 20 distributed software execution there is no motivation to provide an acknowledgment prior to the execution of the command because the command executes quickly and most commands are sequential in nature. In other words, the execution of the next command is dependent upon proper execution of the prior command so there would be no motivation to provide an acknowledgment prior to its actual execution. It is to be understood that other devices, such as digital devices, may be controlled in a manner as described for model railroads.

Referring to FIG. 2, the client program 14 sends a command over the communications transport 12 that is received by an asynchronous command processor 100. The asynchronous command processor 100 queries a local database storage 102 to determine if it is necessary to package 35 a command to be transmitted to a command queue 104. The local database storage 102 primarily contains the state of the devices of the model railroad, such as for example, the speed of a train, the direction of a train, whether a draw bridge is up or down, whether a light is turned on or off, and the 40 configuration of the model railroad layout. If the command received by the asynchronous command processor 100 is a query of the state of a device, then the asynchronous command processor 100 retrieves such information from the local database storage 102 and provides the information to 45 an asynchronous response processor 106. The asynchronous response processor 106 then provides a response to the client program 14 indicating the state of the device and releases the communications transport 12 for the next command

The asynchronous command processor 100 also verifies, 50 using the configuration information in the local database storage 102, that the command received is a potentially valid operation. If the command is invalid, the asynchronous command processor 100 provides such information to the asynchronous response processor 106, which in turn returns 55 an error indication to the client program 14.

The asynchronous command processor 100 may determine that the necessary information is not contained in the local database storage 102 to provide a response to the client program 14 of the device state or that the command is a valid action. Actions may include, for example, an increase in the train's speed, or turning on/off of a device. In either case, the valid unknown state or action command is packaged and forwarded to the command queue 104. The packaging of the command may also include additional information from the 65 local database storage 102 to complete the client program 14 request, if necessary. Together with packaging the command

for the command queue 104, the asynchronous command processor 100 provides a command to the asynchronous request processor 106 to provide a response to the client program 14 indicating that the event has occurred, even though such an event has yet to occur on the physical

railroad layout.

As such, it can be observed that whether or not the command is valid, whether or not the information requested by the command is available to the asynchronous command processor 100, and whether or not the command has executed, the combination of the asynchronous command processor 100 and the asynchronous response processor 106 both verifies the validity of the command and provides a response to the client program 14 thereby freeing up the communications transport 12 for additional commands. Without the asynchronous nature of the resident external controlling interface 16, the response to the client program 14 would be, in many circumstances, delayed thereby resulting in frustration to the operator that the model railroad is performing in a slow and painstaking manner. In this manner, the railroad operation using the asynchronous interface appears to the operator as nearly instantaneously responsive.

Each command in the command queue 104 is fetched by a synchronous command processor 110 and processed. The synchronous command processor 110 queries a controller database storage 112 for additional information, as necessary, and determines if the command has already been executed based on the state of the devices in the controller database storage 112. In the event that the command has already been executed, as indicated by the controller database storage 112, then the synchronous command processor 110 passes information to the command queue 104 that the command has been executed or the state of the device. The asynchronous response processor 106 fetches the information from the command cue 104 and provides a suitable response to the client program 14, if necessary, and updates the local database storage 102 to reflect the updated status of the railroad layout devices.

If the command fetched by the synchronous command processor 110 from the command queue 104 requires execution by external devices, such as the train engine, then the command is posted to one of several external device control logic 114 blocks. The external device control logic 114 processes the command from the synchronous command processor 110 and issues appropriate control commands to the interface of the particular external device 116 to execute the command on the device and ensure that an appropriate response was received in response. The external device is preferably a digital command control device that transmits digital commands to decoders using the train track. There are several different manufacturers of digital command stations, each of which has a different set of input commands, so each external device is designed for a particular digital command station. In this manner, the system is compatible with different digital command stations. The digital command stations 18 of the external devices 116 provide a response to the external device control logic 114 which is checked for validity and identified as to which prior command it corresponds to so that the controller database storage 112 may be updated properly. The process of transmitting commands to and receiving responses from the external devices 116 is slow.

The synchronous command processor 110 is notified of the results from the external control logic 114 and, if appropriate, forwards the results to the command queue 104. The asynchronous response processor 100 clears the results

from the command queue 104 and updates the local database storage 102 and sends an asynchronous response to the client program 14, if needed. The response updates the client program 14 of the actual state of the railroad track devices, if changed, and provides an error message to the client program 14 if the devices actual state was previously improperly reported or a command did not execute properly.

The use of two separate database storages, each of which is substantially a mirror image of the other, provides a performance enhancement by a fast acknowledgement to the 10 client program 14 using the local database storage 102 and thereby freeing up the communications transport 12 for additional commands. In addition, the number of commands forwarded to the external device control logic 114 and the external devices 116, which are relatively slow to respond, 15 is minimized by maintaining information concerning the state and configuration of the model railroad. Also, the use of two separate database tables 102 and 112 allows more efficient multi-threading on multi-processor computers.

In order to achieve the separation of the asynchronous and  $\ ^{20}$ synchronous portions of the system the command queue 104 is implemented as a named pipe, as developed by Microsoft for Windows. The queue 104 allows both portions to be separate from each other, where each considers the other to be the destination device. In addition, the command queue maintains the order of operation which is important to proper operation of the system.

The use of a single command queue 104 allows multiple instantrations of the asynchronous functionality, with one for each different client. The single command queue 104 also allows the sharing of multiple devices, multiple clients to communicate with the same device (locally or remote) in a controlled manner, and multiple clients to communicate with different devices. In other words, the command-queue 104 permits the proper execution in the cases of: (1) one client to many devices, (2) many clients to one device, and (3) many clients to many devices.

The present inventor came to the realization that the digital command stations provided by the different vendors 40 have at least three different techniques for communicating with the digital decoders of the model railroad set. The first technique, generally referred to as a transaction (one or more operations), is a synchronous communication where a command is transmitted, executed, and a response is received 45 therefrom prior to the transmission of the next sequentially received command. The DCS may execute multiple commands in this transaction. The second technique is a cache with out of order execution where a command is executed and a response received therefrom prior to the execution of 50 the next command, but the order of execution is not necessarily the same as the order that the commands were provided to the command station. The third technique is a local-area-network model where the commands are transmitted and received simultaneously. In the LAN model there 55 is no requirement to wait until a response is received for a particular command prior to sending the next command. Accordingly, the LAN model may result in many commands being transmitted by the command station that have yet to be executed. In addition, some digital command stations use 60 two or more of these techniques.

With all these different techniques used to communicate with the model railroad set and the system 10 providing an interface for each different type of command station, there exists a need for the capability of matching up the responses 65 from each of the different types of command stations with the particular command issued for record keeping purposes.

Without matching up the responses from the command stations, the databases can not be updated properly.

Validation functionality is included within the external device control logic 114 to accommodate all of the different types of command stations. Referring to FIG. 3, an external command processor 200 receives the validated command from the synchronous command processor 110. The external command processor 200 determines which device the command should be directed to, the particular type of command it is, and builds state information for the command. The state information includes, for example, the address, type, port, variables, and type of commands to be sent out. In other words, the state information includes a command set for a particular device on a particular port device. In addition, a copy of the original command is maintained for verification purposes. The constructed command is forwarded to the command sender 202 which is another queue, and preferably a circular queue. The command sender 202 receives the command and transmits commands within its queue in a repetitive nature until the command is removed from its queue. A command response processor 204 receives all the commands from the command stations and passes the commands to the validation function 206. The validation function 206 compares the received command against potential commands that are in the queue of the command sender 202 that could potentially provide such a result. The validation function 206 determines one of four potential results from the comparison. First, the results could be simply bad data that is discarded. Second, the results could be partially executed commands which are likewise normally discarded. Third, the results could be valid responses but not relevant to any command sent. Such a case could result from the operator manually changing the state of devices on the model railroad or from another external device, assuming a shared interface to the DCS. Accordingly, the results are validated and passed to the result processor 210. Fourth, the results could be valid responses relevant to a command sent. The corresponding command is removed from the command sender 202 and the results passed to the result processor 210. The commands in the queue of the command sender 202, as a result of the validation process 206, are retransmitted a predetermined number of times, then if error still occurs the digital command station is reset, which if the error still persists then the command is removed and the operator is notified of the error.

#### APPLICATION PROGRAMMING INTERFACE

Train ToolsTM Interface Description Building your own visual interface to a model railroad Copyright 1992-1998 KAM Industries. Computer Dispatcher, Engine Commander, The Conductor, Train Server, and Train Tools are Trademarks of KAM Industries, all Rights Reserved. Questions concerning the product can be EMAILED to: traintools@kam.rain.com You can also mail questions to: KAM Industries

2373 NW 185th Avenue Suite 416 Hillsboro, Oregon 97124 FAX - (503) 291-1221

Table of conter

- OVERVIEW
- 1.1 System Architecture
- TUTORIAL
- Visual BASIC Throttle Example Application Visual BASIC Throttle Example Source Code
- IDL COMMAND REFERENCE

10

	-continued		-continued
	APPLICATION PROGRAMMING INTERFACE		APPLICATION PROGRAMMING INTERFACE
3.1 3.2 3.3	Introduction Data Types Commands to access the server configuration variable database KamCVGetValue KamCVPutValue KamCVPtuEnable KamCVPtuEnable KamCVGetName KamCVGetName	5	communication port KamPortPutConfig KamPordGetConfig KamPortGetName KamPortPutMapController
3.4	KamCVGetMaxRegister Commands to program configuration variables		KamCmdConnect KamCmdDisConnect
3.4	KamProgram KamProgramGetMode KamProgramGetStatus KamProgramReadCV KamProgramReadCV KamProgramReatCV KamProgramReatCN KamProgramReatCN KamProgramReatCN KamProgramReatCN KamProgramReatCN KamProgramBeatCN KamProgramBeatCN KamProgramBecoderFromDataBase	15	5 KamCmdCommand 3.11 Cab Control Commands KamCabGetMessage KamCabPutMessage KamCabGetCabAddr KamCabPutAddrToCab 3.12 Miscellaneus Commands
3.5	Commands to control all decoder types KamDecoderGetMaxModels KamDecoderGetModelName KamDecoderGetModelToObj KamDecoderGetMaxAddress KamDecoderChangeOldNewAddr KamDecoderChangeOldNewAddr KamDecoderGetPort KamDecoderGetPort KamDecoderGetAddrInUse KamDecoderGetModelFromObj KamDecoderGetModelFacility KamDecoderGetObjCount KamDecoderGetObjCount KamDecoderGetVoltGetModelFacility KamDecoderPutAdd KamDecoderPutAdd KamDecoderPutDel KamDecoderGetMDel KamDecoderGetMDel KamDecoderGetMDel KamDecoderGetMDel KamDecoderGetMfgName	25	KamMiscGetErrorMsg KamMiscGetClockTime KamMiscGetClockTime KamMiscGetInterfaceVersion KamMiscGetInterfaceVersion KamMiscGetControllerName KamMiscGetControllerNameAtPort KamMiscGetCommandStationValue KamMiscGetCommandStationValue KamMiscGetCommandStationIndex KamMiscGetCommandStationIndex KamMiscGetControllerID KamMiscGetControllerID KamMiscGetControllerFacility  I. OVERVIEW This document is divided into two sections, the Tutorial, and the IDL Command Reference. The tutorial shows the complete code for a simple Visual BASIC program
	KamDecoderGetPowerMode KamDecoderGetMaxSpeed		that controls all the major functions of a locomotive.  This program makes use of many of the commands described
3.6	Commands to control locomotive decoders  KamEngGetSpeed  KamEngPutSpeed  KamEngGetSpeedSteps  KamEngGetSpeedSteps  KamEngGetFunction  KamEngGetFunction  KamEngGetFunctionMax  KamEngGetFunctionMax  KamEngGetFunctionMax  KamEngGetFunctionName  KamEngGetFunctionName  KamEngGetFunctionName  KamEngGetConsistMax  KamEngGetConsistMax  KamEngGutConsistParent  KamEngPutConsistChild  KamEngPutConsistRemoveObj  Commands to control accessory decoders	40	describes each command in detail.  I. TUTORIAL  A. Visual BASIC Throttle Example Application The following application is created using the Visual BASIC source code in the next section. It controls all major locomotive functions such as expend
3.7	KamAccGetFunction KamAccGetFunctionAll KamAccPutFunctionAll KamAccPutFunctionAll KamAccGetFunctionMax KamAccGetFunctionMax KamAccGetName KamAccGetFunctionName KamAccPutFunctionName KamAccPutFunctionName KamAccPutFunctionName KamAccRegFeedback KamAccPutFeedback KamAccDetFeedback KamOptPutStartStation KamOptPutStartStation KamOptPutStopStation KamOptPutFowerOn	55	SAuthor: \$  \$Revision: \$  \$1.0g: \$  Engine Commander, Computer Dispatcher, Train Server, Train Tools, The Conductor and kamind are registered Trademarks of KAM Industries. All rights reserved.  This first command adds the reference to the Train ServerT Interface object Dim EngCmd As New EngComffc  Engine Commander uses the term Ports, Devices and Controllers Ports -> These are logical ids where Decoders are assigned to. Train ServerT Interface supports a limited number of logical ports. You can also think of ports as mapping to a command station type. This allows you to move decoders between command station without losing any information about the decoder
	KamOprPutPowerOff KamOprPutHardReset KamOprPutEmergencyStop KamOprGetStationStatus	65	Devices -> These are communications channels configured in your computer.  You may have a single device (com1) or multiple devices

11
-continued

12
-continued

	-continued	_	-continued		
	APPLICATION PROGRAMMING INTERFACE	_	APPLICATION PROGRAMMING INTERFACE	_	
maj	OM 1 - COM8, LPT1, Other). You are required to o a port to a device to access a command station. vices start from ID 0 -> max id (FYI; devices do	5	LENZ_1x 2 // Lenz serial support module LENZ_2x 3 // Lenz serial support module DIGIT_DT200 4 // Digitrax direct drive	_	
' che	necessarily have to be serial channel. Always ck the name of the device before you use it as		support using DT200 DIGIT_DCS100 5 // Digitrax direct drive		
' The	l as the maximum number of devices supported.  Command	10	support using DCS100 MASTERSERIES 6 // North Coast engineering		
' lPa	cmd.KamPortGetMaxPhysical(IMaxPhysical, ISerial, ratlel) provides means that IMaxPhysical = rial + IParallel + lOther		master Series 'SYSTEMONE 7 // System One 'RAMFIX 8 // RAMFIxx system		
· Cor	ntroller - These are command the command station		DYNATROL 9 // Dynatrol system Northcoast binary 10 // North Coast binary		
' No	theoast, EasyDCC, Marklin It is recommend	15	'SERIAL 11 // NMRA Serial interface		
that use	tyou check the command station ID before you it.		'EASYDCC 12 // NMRA Serial interface 'MRK6050 13 // 6050 Marklin interface (AC and DC)		
Err	ors - All commands return an error status. If the error value is non zero, then the		'MRK6023 14 // 6023 Marklin hybrid interface (AC)		
1	other return arguments are invalid. In	20	*ZTC 15 // ZTC Systems ltd		
•	general, non zero errors means command was		DIGIT_PR1 16 // Digitrax direct drive		
	not executed. To get the error message, you need to call KamMiscErrorMessage and		support using PR1 'DIRECT 17 // Direct drive interface routine		
,	supply the error number		10ttttc	*	
	Operate your layout you will need to perform a	25	iLogicalPort = 1 'Select Logical port 1 for		
' (ph	pping between a Port (logical reference), Device sysical communications channel) and a Controller mmand station) for the program to work. All		communications iController = 1 'Select controller from the list above.		
' refe	rences uses the logical device as the reference		iComPort = 0 'use COM1; 0 means com1 (Digitrax must use Com1 or Com2)		
, 400	race for access.	30	Digitrax Baud rate requires 16.4K!		
	dresses used are an object reference. To use an		'Most COM ports above Com2 do not		
	lress you must add the address to the command		'support 16.4K. Check with the		
	tion using KamDecoderPutAdd One of the return ues from this operation is an object reference		'manufacture of your smart com card 'for the baud rate. Keep in mind that		
	t is used for control.		Dumb com cards with serial port		
,	• • • • • • • • • • • • • • • • • • • •	35	support Com1 - Com4 can only support		
	need certain variables as global objects; since		'2 com ports (like com1/com2		
	information is being used multiple times calPort, iController, iComPort		'or com3/com4) 'If you change the controller, do not		
	Rate, iPortParity, iPortStop, iPortRetrans,		forget to change the baud rate to		
	rtWatchdog, iPortFlow, iPortData		match the command station. See your		
	neObject As Long, iDecoderClass As Integer,	40	'user manual for details		
	ecoderType As Integer Controller As Long		'0: // Baud rate is 300		
	Logical As Long, IMaxPhysical As Long, IMaxSerial		1: // Baud rate is 1200		
	Long, lMaxParallel As Long		2: // Baud rate is 2400		
*******	**********		'3: // Baud rate is 4800		
Form load	the initial buttons	45	' 4: // Baud rate is 9600 ' 5: // Baud rate is 14.4		
	nterface information		' 6: // Baud rate is 16.4		
******	<b>非新生素素素 医香茶香香香香香香香香香香香香香</b>		'7: // Baud rate is 19.2		
	b Form_load()		iPortRate = 4		
Di	m strVer As String, strCom As String, strCntrl As String		' Parity values 0-4 -> no, odd, even, mark, space		
	m iError As Integer	50	iPortParity = 0		
	et the interface version information		Stop bits 0,1,2 -> 1, 1.5, 2		
	tButtonState (False)  ror = EngCmd.KamMiscGetInterfaceVersion(strVer)		iPortStop = 0 iPortRetrans = 10		
	(iError) Then		iPortWatchdog = 2048		
	MsgBox (("Train Server not loaded. Check		iPortFlow = 0		
	DCOM-95")) iLogicalPort = 0	55	Data bits 0 - > 7 Bits, 1-> 8 bits iPortData = 1		
	LogPort.Caption = iLogicalPort		Display the port and controller information		
	ComPort.Caption = "???"		iError = EngCmd.KamPortGetMaxLogPorts(lMaxLogical)		
E21.	Controller.Caption = "Unknown"		iError = EngCmd.KamPortGetMaxPhysical(lMaxPhysical, lMaxSerial, lMaxParallel)		
Els	se MsgBox (("Simulation(COM1) Train Server " &	a	'Get the port name and do some checking		
	strVcr))	60	iError = EngCmd.KamPortGetName(iComPort, strCom) SetError (iError)		
	'Configuration information; Only need to		If (iComPort > lMaxSerial) Then MsgBox ("Com port		
	change these values to use a different controller		our of range") iError =		
	·		EngCmd.KamMiscGetControllerName(iController,		
	UNKNOWN 0 // Unknown control type	65	ui 2111)		
	SIMULAT 1 // Interface simulator		If (iLogicalPort > lMaxLogical) Then MsgBox		

13

14

iStatus = EngCmd.KamDecoderPutAdd(Address.Text, iLogicalPort, iLogicalPort, 0,

iDecoderType, lEngineObject)

```
-continued
                                                  -continued
                     APPLICATION PROGRAMMING INTERFACE
                                                                                                                                                        APPLICATION PROGRAMMING INTERFACE
                                                                                                                                              iError = EngCmd.KamPortPutConfig(iLogicalPort, 4
("Logical port out of range")
                                                                                                                                             iError = EngCmd.KamrotirutConing(LlogicalFort, 40
iPortWatchdog, 0) * setting PORT_WATCHDOG
iError = EngCmd.KamPortPutConfig(LlogicalPort, 5,
iPortFlow, 0) * setting PORT_FLOW
iError = EngCmd.KamPortPutConfig(ILogicalPort, 6,
iPortData, 0) * setting PORT_DATABITS
                        SetError (iError)
        End If
            'Display values in Throttle.
            LogPort.Caption = iLogicalPort
            ComPort.Caption = strCom
            Controller.Caption = strCntrl
                                                                                                                            10 'We need to set the appropriate debug mode for display.
'this command can only be sent if the following is true
End Sub
                                                                                                                                     -Controller is not connected
'Send Command
                                                                                                                                     -port has not been mapped
                                                                                                                                     -Not share ware version of application (Shareware
'Note:
                                                                                                                                              always set to 130)
            Please follow the command order. Order is important
                                                                                                                                    Write Display Log
File Win Level
            for the application to work!
                                                                                                                                                                           Debug
                                                                                                                                                                         Value
                                                                                                                                     1 + 2 + 4 = 7
                                                                                                                                                                                 -> LEVEL1 -- put packets into
Private Sub Command_Click()
            'Send the command from the interface to the command
                                                                                                                                              queues
            station, use the engineObject
Dim iError, iSpeed As Integer
If Not Connect.Enabled Then
                                                                                                                                    1 + 2 + 8 = 11
                                                                                                                                                                                 -> LEVEL2 -- Status messages
                                                                                                                                              send to window
                                                                                                                                                                                 -> LEVEL 3 --
                                                                                                                                        +2+16=19
                        TrainTools interface is a caching interface
                                                                                                                                    1 + 2 + 32 = 35
                                                                                                                                                                                 -> LEVEL4 -- All system
                        This means that you need to set up the CV's or other operations first; then execute the
                                                                                                                                              semaphores/critical sections
                                                                                                                                    1 + 2 + 64 = 67
                                                                                                                                                                                 -> LEVEL5 -- detailed
                                                                                                                                    debugging information
1 + 2 + 128 = 0 131
                         'command.
                                                                                                                                                                                 -> COMMONLY -- Read comm write
                        iSpeed - Speed.Text
                        iError =
            EngCmd.KamEngPutFunction(lEngineObject, 0, F0.Value)
                                                                                                                                    You probably only want to use values of 130. This will
                        iError -
                         EngCmd.KamEngPutFunction(IEngineObject, 1,
                                                                                                                                    give you a display what is read or written to the
                        F1. Value)
                                                                                                                                    controller. If you want to write the information to
                                                                                                                                    'disk, use 131. The other information is not valid for
                        iError =
                         EngCmd.KamEngPutFunction(lEngineObject, 2,
                                                                                                                                    end users
                                                                                                                                                              This does effect the performance of you
                        F2. Value)
                                                                                                                                   ' Note: 1.
                                                                                                                                                              system; 130 is a save value for debug
                        iError =
                                                                                                                                                              display. Always set the key to 1, a value of 0 will disable debug
                         EngCmd.KamEngPutFunction(lEngineObject, 3,
                        F3. Value)
iError = EngCmd.KamEngPutSpeed(lEngineObject,
                                                                                                                                                              The Digitrax control codes displayed are
                         iSpeed, Direction. Value)
                                                                                                                                                              encrypted. The information that you
                                                                                                                                                              determine from the control codes is that
                        If iError = 0 Then iError =
                                                                                                                            35
                                                                                                                                                              information is sent (S) and a response is
                         EngCmd.KamCmdCommand(lEngineObject)
                         SetError (iError)
                                                                                                                                                               received (R)
                     End If
 End Sub
                                                                                                                                    iDebugMode = 130
                                                                                                                                   iValue = Value.Text' Display value for reference
iError = EngCmd.KamPortPutConfig(iLogicalPort, 7, iDebug,
iValue) setting PORT_DEBUG
'Now map the Logical Port, Physical device, Command
station and Controller
             ***********
 'Connect Controller
 Private Sub Connect_Click()
            Dim iError As Integer
              "These are the index values for setting up the port
                                                                                                                                    iError = EngCmd.KamPortPutMapController(iLogicalPort,
                                                                                                                                    iController, iComPort)
iError = EngCmd.KamCmdConnect(iLogicalPort)
              PORT_RETRANS
                                                               0 // Retrans index
              PORT_RATE
                                                                1 // Retrans index
                                                                                                                                   iError = EngCmd.KamOprPutTurnOnStation(iLogicalPort)
                                                                                                                                   If (iError) Then
SetButtonState (False)
              PORT_PARITY
                                                                2 // Retrans index
              PORT_STOP
                                                                3 // Retrans index
             PORT_WATCHDOG
PORT_FLOW
PORT_DATABITS
                                                                4 // Retrans index
                                                                                                                                           Else
                                                                                                                                               SetButtonState (True)
                                                               5 // Retrans index
                                                                                                                                            End If
                                                                6 // Retrans index
               PORT_DEBUG
                                                                7 // Retrans index
                                                                                                                             50 SetError (iError) 'Displays the error message and error
              PORT_PARALLEL
                                                                8 // Retrans index
                                                                                                                                               number
                                                                                                                                   End Sub
                         These are the index values for setting up the
                         port for use
             'PORT_RETRANS
'PORT_RATE
                                                                0 // Retrans index
                                                                                                                                    'Set the address button
                                                                1 // Retrans index
              PORT_PARTTY
                                                                2 // Retrans index
                                                                                                                             55 Private Sub DCCAddr_Click()
              PORT_STOP
PORT_WATCHDOG
                                                               3 // Retrans index
4 // Retrans index
                                                                                                                                              Dim iAddr, iStatus As Integer
'All addresses must be match to a logical port to
              PORT_FLOW
                                                                5 // Retrans index
                                                                                                                                               PORT_DATABITS
PORT_DEBUG
                                                                6 // Retrans index
7 // Retrans index
                                                                                                                                                                                          ' Set the decoder type to an NMRA
               PORT_PARALLEL
                                                                8 // Retrans index
             'PORT_PARALIEL 8 // Retrans index iError = EngCmd.KamPortPutConfig(iLogicalPort, 0, iPortRetrans, 0) 'setting PORT_RETRANS iError = EngCmd.KamPortPutConfig(iLogicalPort, 1 iPortRate, 0) 'setting PORT_RATE iError = EnqCmd.KamPortPutConfig(iLogicalPort, 2, iPortParity, 0) 'setting PORT_PARITY iError = EngCmd.KamPortPutConfig(iLogicalPort, 3 iPortParity, 0) 'setting PORT_PARITY iError = EngCmd.KamPortPutConfig(iLogicalPort, 3 iPortParity, 0) 'setting PORT_STANDARD (See 1) (See
                                                                                                                             60
                                                                                                                                                Engine and Accessory
                                                                                                                                               Once we make a connection, we use the lEngineObject as the reference object to send control information
                                                                                                                                                If (Address.Text > 1) Then
```

65

iPortStop, 0) 'setting PORT\_STOP

15
-continued

16
-continued

```
APPLICATION PROGRAMMING INTERFACE
                                                                                               APPLICATION PROGRAMMING INTERFACE
       SetError (iStatus)
                                                                                 Private Sub ONCmd_Click()
       If(IEngineObject) Then
                                                                                         Dim iError As Integer
            Command.Enabled = True 'turn on the control
                                                                                         iError = EngCmd.KamOprPutPowerOn(iLogicalPort)
            (send) button
            Throttle.Enabled = True Turn on the throttle
                                                                                         SetError (iError)
                                                                                 End Sub
            MsgBox ("Address not set, check error message")
                                                                             10
                                                                                  Throttle slider control
            End If
                                                                                 Private Sub Throttle_Click()
            MsgBox ("Address must be greater then 0 and
                                                                                         If (lEngineObject) Then
                 less then 128")
                                                                                             If (Throttle. Value > 0) Then
                                                                                                Speed.Text = Throttle.Value
End If
End Sub
                                                                             15
Disconenct button
                                                                                              End If
                                                                                  End Sub
                                                                                         IDL COMMAND REFERENCE
Private Sub Disconnect_Click()
                                                                                         A. Introduction
       Dim iError As Integer
       iError = EngCmd.KamCmdDisConnect(iLogicalPort)
                                                                                              This document describes the IDL interface to
       SetError (iError)
                                                                                 the KAM Industries Engine Commander Train Server. The
                                                                                 Train Server DCOM server may reside locally or on a
network node This server handles all the background
       SetButtonState (False)
End Sub
                                                                                  details of controlling your railroad. You write simple,
'Display error message
                                                                                  front end programs in a variety of languages such as
                                                                                 BASIC, Java, or C++ to provide the visual interface to the user while the server handles the details of
Private Sub SetError(iError As Integer)
                                                                                  communicating with the command station, etc.

A. Data Types
       Dim szError As String
       Dim iStatus
                                                                                  Data is passed to and from the IDL interface using a
        This shows how to retrieve a sample error message
                                                                                  several primitive data types. Arrays of these simple types are also used. The exact type passed to and from
       from the interface for the status received.
       iStatus = EngCmd.KamMiscGetErrorMsg(iError, szError)
                                                                                 your program depends on the programming language your are
       ErrorMsg.Caption = szError
                                                                                  using.

The following primitive data types are used:
        Result.Caption = Str(iStatus)
End Sub
                                                                                  IDL Type BASIC Type C++ Type Java Type
                                                                                                                                     Description
'Set the Form button state
                                                                                  short
                                                                                              short
                                                                                                            short
                                                                                                                         short
                                                                                                                                     Short signed integer
                                                                                                                                     Signed integer
                                                                                              int
                                                                                                             int
                                                                                                                         int
                                                                                                                                     Text string
Unsigned 32 bit value
                                                                                  BSTR
                                                                                              BSTR
                                                                                                             BSTR
                                                                                                                         BSTR
Private Sub SetButtonState(iState As Boolean)
                                                                             35
                                                                                  long long long
Name ID CV Range Valid CV's
        'We set the state of the buttons; either connected
                                                                                                                         long
                                                                                                                                    Address Range
                                                                                                                      Functions
        or disconnected
       If (iState) Then
                                                                                  Steps
            Connect.Enabled = False
Disconnect.Enabled = True
                                                                                                            None
                                                                                  NMRA Compatible
                                                                                                      0
                                                                                                                      None
                                                                                                                                     1-99
                                                                                                            1-8
                                                                                                                                     1-127
                                                                                                                      1-8
                                                                                  Baseline
             ONCmd.Enabled = True
                                                                                  Extended
                                                                                                            1-106
                                                                                                                      1-9, 17, 18, 19, 23, 24, 29, 30,
             OffCmd.Enabled = True
                                                                                  40 66-05
                                                                                                   9
                                                                                                             1-10239
                                                                                                                             14,28,128
9 1-1
                                                                                                                      1-106
                                                                                                                                    1-10239
                                                                                                                                                 14,28,128
             DCCAddr.Enabled = True
                                                                                  All Mobile
                                                                                                       3
                                                                                                            1-106
                                                                                                CV Range
             UpDownAddress.Enabled = True
                                                                                  Name ID
                                                                                                               Valid CV's
                                                                                                                                Functions Address Range
       'Now we check to see if the Engine Address has been
                                                                                                               513-593
                                                                                                                              513-593 8
                                                                                                                             513-1024 8
       'set; if it has we enable the send button If (lEngineObject > 0) Then
                                                                                                               513-1024
                                                                                  All Stationary
                                                                                  A long /DecoderObject/D value is returned by the
             Command.Enabled = True
                                                                                  KamDecoderPutAdd call if the decoder is successfully
                                                                                  registered with the server. This unique opaque ID should
             Throttle.Enabled = True
                                                                                  be used for all subsequent calls to reference this
          Else
             Command.Enabled = False
                                                                                  decoder.
             Throttle.Enabled = False
                                                                                         Commands to access the server configuration variable
          End If
       Else
                                                                                              This section describes the commands that access
                                                                                  the server configuration variables (CV) database. These
             Connect.Enabled = True
             Disconnect.Enabled = False
                                                                                  CVs are stored in the decoder and control many of its
                                                                                  characteristics such as its address. For efficiency, a copy of each CV value is also stored in the server
             Command Enabled = False
             ONCmd.Enabled = False
                                                                                  database. Commands such as KamCVGetValue and KamCVPutValue communicate only with the server, not the
             OffCmd.Enabled = False
             DCCAddr.Enabled = False
             UpDownAddress.Enabled = False
                                                                                  actual decoder. You then use the programming commands in
             Throttle.Enabled = False
                                                                                  the next section to transfer CVs to and from the decoder.
             End If
                                                                                  0KamCVGetValue
                                                                                                                         Direction
End Sub
                                                                                  Parameter List
                                                                                                               Range
                                                                                  Parameter List -7F | IDecoderObjectID long 1 | 1-1024 2
                                                                                                      Type
                                                                                                                                     Description
                                                                                                                         In
                                                                                                                                     Decoder object ID
Power Off function
                                                                                                                                     CV register
                                                                                                                         Ĭn
                                                                                                                                     Pointer to CV value
                                                                                  pCV Value
Private Sub OffCmd_Click()
                                                                                         Opaque object ID handle returned by
                                                                                   KamDecoderPutAdd.
        Dim iError As Intege
        iError = EngCmd.KamOprPutPowerOff(iLogicalPort)
                                                                                         Range is 1-1024. Maximum CV for this decoder is
        SetError (iError)
                                                                                   given by KamCVGetMaxRegister.
                                                                                         CV Value pointed to has a range of 0 to 255.
End Sub
                                                                                  Return Value
                                                                                                      Туре
                                                                                                                    Range
                                                                                                                                     Description
Power On function
                                                                                   iError short
                                                                                                                    Error flag
```

**17** 

18

# -continued -continued

APPLICATION PROGRAMMING INTERFACE			NTERFACE	APPLICATION PROGRAMMING INTERFACE						
iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg). KamCVGetValue takes the decoder object ID and configuration variable (CV) number				tes the	5	pbsCVNameString to the name of the CV as defined in NMRA Recommended Practice RP 9.2.2.				
as parameters. It se	ets the me	mory poi	nted to by p	CVValue		OKamCVGetMinRegister Parameter List Type Range Direction Description				
to the value of the variable.	server co	py of the	configuratio	n		IDecoderObjectID long 1 In Decoder object ID pMinRegister int * 2 Out Pointer to min CV				
0KamCVPutValue					10	register number				
Parameter List IDecoderObjectID	Type long	Range 1	Direction In	Description Decoder object ID		1 Opaque object ID handle returned by KamDecoderPutAdd.				
iCVRegint	1-1024	2	In	CV register		2 Normally 1-1024. 0 on error or if decoder does not				
iCVValue  1 Opaque obj	int ject ID ha	0-255 indle retur	In ned by	CV value		support CVs.  Return Value Type Range Description				
KamDecoderPutAc	dd.		-		15	iError short 1 Error flag				
2 Maximum given by KamCVO			ium CV ior	this decoder is		1 iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg).				
Return Value iError short	Type 1	Rai	nge or flag	Description		KamCVGetMinRegister takes a decoder object ID as a parameter. It sets the memory pointed to by pMinRegister				
			o is an error	number		to the minimum possible CV register number for the				
(see KamMiscGettl KamCVPutValue t			niect ID. con	figuration	20	specified decoder.  0KamCVGetMaxRegister				
variable (CV) num						Parameter List Type Range Direction Description				
It sets the server c iCVValue.	opy of the	e specified	decoder C	/ to		IDecoderObjectID   long 1   In   Decoder object ID				
0KamCVGetEnabl	e					register number				
Parameter List IDecoderObjectID	Type	Range 1	Direction In	Description Decoder object ID	25	1 Opaque object ID handle returned by KamDecoderPutAdd.				
iCVRegint	1-1024	2	In	CV number		Normally 1-1024. 0 on error or if decoder does not				
pEnable 1 Opaque ob	int *	3 Indle retur	Out ned by	Pointer to CV bit mask		support CVs.  Return Value Type Range Description				
KamDecoderPutA	dd.		•			iError short 1 Error flag				
2 Maximum given by KamCVO			num CV for	this decoder is	30	1 iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg).				
3 0x0001 - SET	r_cv_i	VUSE (		C_CV_READ_DIRTY		KamCVGetMaxRegister takes a decoder object ID as a				
0x0004 - SET SET_CV_E			IRIY Oxo	008 -		parameter. It sets the memory pointed to by pMaxRegister to the maximum possible CV register number for the				
0x0010 - SET	r_cv_e	RROR_V		B 1.0		specified decoder.				
Return Value iError short	Type 1	Rar Err	nge or flag	Description	35	A. Commands to program configuration variables  This section describes the commands read and				
			o is an error		55	write decoder configuration variables (CVs). You should				
(see KamMiscGetl decoder object ID,						initially transfer a copy of the decoder CVs to the server using the KamProgramReadDecoderToDataBase command.				
and a pointer to store the enable flag as parameters. It sets the location pointed to by pEnable.					You can then read and modify this server copy of the CVs.					
0KamCVPutEnabl		oy pichao.	ic.		40	Finally, you can program one or more CVs into the decoder using the KamProgramCV or KamProgramDecoderFromDataBase				
Parameter List iDecoderObjectID	Type	Range 1	Direction In	Description Decoder object ID	70	command. Not that you must first enter programming mode by issuing the KamProgram command before any programming				
iCVRegint	1-1024	2	In	CV number		can be done.				
iEnableint 1 Opaque ob	iect ID ha	3 indle retui	In ned by	CV bit mask		0KamProgram Parameter List Type Range Direction Description				
KamDecoderPutA	dd.				45	lDecoderObjectID long 1 In Decoder object ID				
2 Maximum given by KamCVO			num CV tor	this decoder is	45	iProgLogPort int 1-65535 2 In Logical programming				
3 0x0001 - SET	[_CV_I	VUSE (		C_CV_READ_DIRTY		port ID				
0x0004 - SET SET_CV_E			IKI I UXU	008 -		iProgMode int 3 In Programming mode  1 Opaque object ID handle returned by				
0x0010 - SET_CV_ERROR_WRITE				Description	50	KamDecoderPutAdd.				
Return Value Type Range Description iError short 1 Error flag				Description	30	2 Maximum value for this server given by KamPortGetMaxLogPorts.				
1 iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg).				number		3 0 - PROGRAM_MODE_NONE 1 - PROGRAM_MODE_ADDRESS				
KamCVPutEnable takes the decoder object ID, configuration				nfiguration		2 - PROGRAM_MODE_REGISTER				
variable (CV) number, and a new enable state as parameters. It sets the server copy of the CV bit mask				nask	55	3 - PROGRAM_MODE_PAGE 4 - PROGRAM_MODE_DIRECT				
to iEnable.					33	5 - DCODE_PRGMODE_OPS_SHORT				
0KamCVGetName Parameter List	: Туре	Range	Direction	n Description		6 - PROGRAM_MODE_OPS_LONG Return Value Type Range Description				
iCV	int	1-102	24 In	CV number		iError short 1 Error flag				
pbsCVNameString	BSTR	1	Out	Pointer to CV name string		1 iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg)				
			language. It		60	KamProgram take the decoder object ID, logical				
Return Value	Type	nipty stri Rai	ng on error.	Description		programming port ID, and programming mode as parameters.  It changes the command station mode from normal operation				
iError short 1 iError = 0	1 for succes		or flag o is an erroi	number		(PROGRAM_MODE_NONE) to the specified programming mode.  Once in programming modes, any number of programming				
(see KamMiscGet	ErrorMsg)	).				commands may be called. When done, you must call				
KamCVGetName as a parameter. It				CV) number	65	KamProgram with a parameter of PROGRAM_MODE_NONE to return to normal operation.				
1		-5 6	,			** • • · · ·				

### US 6.530.329 B2

19

APPLICATION PROGRAMMING INTERFACE

20

-continued

APPLICATION PROGRAMMING INTERFACE

### -continued

#### 0KamProgramGetMode KamProgramCV takes the decoder object ID, configuration Parameter List Type variable (CV) number, and a new CV value as parameters. Range Direction Description It programs (writes) a single decoder CV using the 1DecoderObjectID long Decoder object ID 1-65535 specified value as source data. OKamProgramReadDecoderToDataBase int 2 Logical iProgLogPort In programming port ID Parameter List Type Range Direction lDecoderObjectID long int \* 3 Ont Programming mode 1 Ĭn Decoder object ID piProgMode Opaque object ID handle returned by Opaque object ID handle returned by KamDecoderPutAdd. KamDecoderPutAdd. Range Error flag Maximum value for this server given by Return Value Type Description KamPortGetMaxLogPorts. 3 0 - PROGRAM\_MODE\_NONE iError short iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg). KamProgramReadDecoderToDataBase takes the decoder object 1 - PROGRAM\_MODE\_ADDRESS 2 - PROGRAM\_MODE\_REGISTER 3 - PROGRAM\_MODE\_PAGE ID as a parameter. It reads all enabled CV values from 4 - PROGRAM\_MODE\_DIRECT the decoder and stores them in the server database. 0KamProgramDecoderFromDataBase 5 - DOODE PROMODE OPS SHORT 6 - PROGRAM\_MODE\_OPS\_LONG Type Range Direction Description IDecoderObjectID long 1 In Opaque object ID handle returned by Return Value Type Range Description Decoder object ID iError short Error flag Description KamDecoderPutAdd. iError = 0 for success. Nonzero is an error number Range Error flag (see KamMiscGetErrorMsg). Return Value Туре Description iError short KamProgramGetMode take the decoder object ID, logical iError = 0 for success. Nonzero is an error number programming port ID, and pointer to a place to store (see KamMiscGetErrorMsg). KamProgramDecoderFromDataBase takes the decoder object ID the programming mode as parameters. It sets the memory pointed to by piProgMode to the present programming mode. 0KamProgramGetStatus as a parameter. It programs (writes) all enabled decoder Parameter List Type Range Direction Description CV values using the server copy of the CVs as source Parameter List 271 IDecoderObjectID long 0-1024 Decoder object ID data. In CV number Commands to control all decoder types piCVAllStatus This section describes the commands that all decoder types. These commands do things such getting the int \* Out Or'd decoder programming status maximum address a given type of decoder supports, adding Opaque object ID handle returned by KamDecoderPutAdd. decoders to the database, etc 0 returns OR'd value for all CVs. Other values 0KamDecoderGetMaxModels Parameter List Type Range Direction Description return status tor just that CV. 0x0001 - SET\_CV\_INUSE 0x0002 - SET\_CV\_READ\_DIRTY piMaxModels Out Pointer to Max model ID 35 Normally 1-65535. 0 on error. 0x0004 - SET\_CV\_WRITE\_DIRTY Return Value 0x0008 - SET\_CV\_ERROR\_READ 0x0010 - SET\_CV\_ERROR\_WRITE Range Туре Description Error flag iError short Return Value iError = 0 for success. Nonzero is an error number Type Range Description Error flag (see KamMiscGetErrorMsg). KamDecoderGetMaxModels takes no parameters. It sets the iError short iError = 0 for success. Nonzero is an error number memory pointed to by piMaxModels to the maximum decoder (see KamMiscGetErrorMsg). type ID. 0KamDecoderGetModelName KamProgramGetStatus take the decoder object ID and pointer to a place to store the OR'd decoder programming status as parameters. It sets the memory pointed to by Parameter List Type 1-65535 Range Direction Description piProgMode to the present programming mode. 0KamProgramReadCV Decoder type ID iModel int In BSTR \* Decoder name pbsModelName Out Type Range Direction Description string lDecoderObjectID long 1 In Decoder object ID Maximum value for this server given by CV number KamDecoderGetMaxModels. iCVRegint In Opaque object ID handle returned by Exact return type depends on language. It is KamDecoderPutAdd. Cstring \* for C++. Empty string on error. Return Value Type Range Maximum CV is 1024. Maximum CV for this decoder is Type Description given by KamCVGetMaxRegister. iError short Error flag Return Value Type Range Error flag Description iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg). KamPortGetModelName takes a iError short iError = 0 for success. Nonzero is an error number decoder type ID and a pointer to a string as parameters (see KamMiscGetErrorMsg). KamProgramCV takes the decoder object ID, configuration It sets the memory pointed to by pbsModelName to a BSTR containing the decoder name. variable (CV) number as parameters. It reads the 0KamDecoderSetModelToObi specified CV variable value to the server database. 0KamProgramCV Type Range Direction int 1 In Parameter List Description Decoder model ID Parameter List Direction Description IDecoderObjectID long In Decoder object ID Type Range Maximum value for this server given by iDecoderObjectID long In Decoder object ID 1 iCVRegint CV number KamDecoderGetMaxModels. In Opaque object ID handle returned by KamDecoderPutAdd. iCVValue int 0-255 In CV value Opaque object ID handle returned by KamDecoderPutAdd. Return Value Туре Range Description Maximum CV is 1024. Maximum CV for this decoder is given by KamCVGetMaxRegister. Return Value Type Range Description Error flag iError short iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg). Error flag iError short KamDecoderSetModelToObj takes a decoder ID and decoder iError = 0 for success. Nonzero is an error number object ID as parameters. It sets the decoder model type (see KamMiscGetErrorMsg). of the decoder at address lDecoderObjectID to the type

21
-continued

22
-continued

	_	-continued
APPLICATION PROGRAMMING INTERFACE	_	APPLICATION PROGRAMMING INTERFACE
specified by iModel.	5	KamPortGetMaxLogPorts.
0KamDecoderGetMaxAddress		3 1 - DECODER_ENGINE_TYPE, 2 DECODER_SWITCH_TYPE
Parameter List Type Range Direction Description		2 - DECODER_SWITCH_TYPE, 3 - DECODER_SENSOR_TYPE.
iModel int 1 In Decoder type ID piMaxAddress int * 2 Out Maximum decoder		Return Value Type Range Description
piMaxAddress int * 2 Out Maximum decoder address		iError short 1 Error flag
	10	
1 Maximum value for this server given by KamDecoderGetMaxModels	10	use. Nonzero is an error number (see
2 Model dependent. 0 returned on error.		KamMiscGetErrorMsg). IDS_ERR_ADDRESSEXIST returned if
Return Value Type Range Description		call succeeded but the address exists.
iError short 1 Error flag		KamDecoderCheckAddrInUse takes a decoder address, logical
1 iError = 0 for success. Nonzero is an error number		port, and decoder class as parameters. It returns zero
(see KamMiscGetErrorMsg).	16	if the address is not in use. It will return
KamDecoderGetMaxAddress takes a decoder type ID and a	13	IDS_ERR_ADDRESSEXIST if the call succeeds but the address
pointer to store the maximum address as parameters. It		already exists. It will return the appropriate non zero
sets the memory pointed to by piMaxAddress to the maximum		error number if the calls fails.
address supported by the specified decoder.		0KamDecoderGetModelFromObj
0KamDecoderChangeOldNewAddr		Parameter List Type Range Direction Description
Parameter List Type Range Direction Description	20	lDecoderObjectID long 1 In Decoder object ID
lOldObjID long 1 In Old decoder object ID	20	piModelint * 1-65535 2 Out Pointer to decoder
iNewAddr int 2 In New decoder address		type ID
plNewObjID long * 1 Out New decoder object II	)	1 Opaque object ID handle returned by
1 Opaque object ID handle returned by		KamDecoderPutAdd.
KamDecoderPutAdd.		2 Maximum value for this server given by
2 1-127 for short locomotive addresses. 1-10239 for		KamDecoderGetMaxModels.
long locomotive decoders. 0-511 for accessory decoders.	25	Return Value Type Range Description
Return Value Type Range Description		iError short 1 Error flag
iError short 1 Error flag		iError = 0 for success. Nonzero is an error number
1 iError = 0 for success. Nonzero is an error number		(see KamMiscGetErrorMsg).
(see KamMiscGetErrorMsg).		KamDecoderGetModelFromObj takes a decoder object ID and
KamDecoderChangeOldNewAddr takes an old decoder object ID		pointer to a decoder type ID as parameters. It sets the
and a new decoder address as parameters. It moves the	30	memory pointed to by piModel to the decoder type ID
specified locomotive or accessory decoder to iNewAddr and		associated with iDCCAddr.
sets the memory pointed to by plNewObjID to the new		0KamDecoderGetModelFacility
object ID. The old object ID is now invalid and should		Parameter List Type Range Direction Description
no longer be used.		IDecoderObjectID long 1 In Decoder object ID
0KamDecoderMovePort		pdwFacility long * 2 Out Pointer to decoder
Parameter List Type Range Direction Description	35	facility mask
IDecoderObjectID long 1 In Decoder object ID	33	1 Opaque object ID handle returned by
iLogicalPortID int 1-65535 2 In Logical port ID		KamDecoderPutAdd.
1 Opaque object ID handle returned by		2 0 - DCODE_PRGMODE_ADDR
KamDecoderPutAdd.		1 - DCODE_PRGMODE_REG
2 Maximum value for this server given by		2 - DCODE_PRGMODE_PAGE
KamPortGetMaxLogPorts.		3 - DCODE_PRGMODE_DIR
Return Value Type Range Description	40	4 - DCODE_PRGMODE_FLYSHT
iError short 1 Error flag		5 - DCODE_PRGMODE_FLYLNG
1 iError = 0 for success. Nonzero is an error number		6 - Reserved
(see KamMiscGetErrorMsg).		7 - Reserved
KamDecoderMovePort takes a decoder object ID and logical		8 - Reserved
port ID as parameters. It moves the decoder specified by		9 - Reserved
IDecoderObjectID to the controller specified by	45	10 - Reserved
iLogicalPortID.		11 - Reserved
0KamDecoderGetPort		12 - Reserved
Parameter List Type Range Direction Description		13 - DCODE_FEAT_DIRLIGHT
lDecoderObjectID long 1 In Decoder object ID		14 - DCODE_FEAT_LNGADDR
piLogicalPortID int * 1-65535 2 Out Pointer to		15 - DCODE_FEAT_CVENABLE
logical port ID	50	16 - DCODE_FEDMODE_ADDR
		17 - DCODE_FEDMODE_REG
1 Opaque object ID handle returned by		
		18 - DCODE_FEDMODE_PAGE
1 Opaque object ID handle returned by		18 - DCODE_FEDMODE_PAGE 19 - DCODE_FEDMODE_DIR
1 Opaque object ID handle returned by KamDecoderPutAdd.		19 - DCODE_FEDMODE_DIR 20 - DCODE_FEDMODE_FLYSHT
1 Opaque object ID handle returned by KamDecoderPutAdd. 2 Maximum value for this server given by KamPortGetMaxLogPorts. Return Value Type Range Description		19 - DCODE_FEDMODE_DIR 20 - DCODE_FEDMODE_FLYSHT 21 - DCODE_FEDMODE_FLYLNG
1 Opaque object ID handle returned by KamDecoderPutAdd. 2 Maximum value for this server given by KamPortGetMaxLogPorts. Return Value Type Range Description iError short 1 Error flag	55	19 - DCODE_FEDMODE_DIR 20 - DCODE_FEDMODE_FLYSHT 21 - DCODE_FEDMODE_FLYLNG Return Value Type Range Description
1 Opaque object ID handle returned by KamDecoderPutAdd. 2 Maximum value for this server given by KamPortGetMaxLogPorts. Return Value Type Range Description iError short 1 Error flag 1 iError = 0 for success. Nonzero is an error number	55	19 - DCODE_FEDMODE_DIR 20 - DCODE_FEDMODE_FLYSHT 21 - DCODE_FEDMODE_FLYLNG  Return Value Type Range Description iError short 1 Error flag
1 Opaque object ID handle returned by KamDecoderPutAdd. 2 Maximum value for this server given by KamPortGetMaxLogPorts. Return Value Type Range Description iError short 1 Error flag 1 iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg).	55	19 - DCODE_FEDMODE_DIR 20 - DCODE_FEDMODE_FLYSHT 21 - DCODE_FEDMODE_FLYLNG  Return Value Type Range Description iError short 1 Error flag 1 iError = 0 for success. Nonzero is an error number
1 Opaque object ID handle returned by KamDecoderPutAdd. 2 Maximum value for this server given by KamPortGetMaxLogPorts. Return Value Type Range Description iError short 1 Error flag 1 iError = 0 for success. Nonzero is an error number (see KamMissGetErrorMsg). KamDecoderMovePort takes a decoder object ID and pointer	55	19 - DCODE_FEDMODE_DIR 20 - DCODE_FEDMODE_FLYSHT 21 - DCODE_FEDMODE_FLYLNG  Return Value Type Range Description iError short 1 Error flag 1 iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg).
1 Opaque object ID handle returned by KamDecoderPutAdd. 2 Maximum value for this server given by KamPortGetMaxLogPorts. Return Value Type Range Description iError short 1 Error flag 1 iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg). KamDecoderMovePort takes a decoder object ID and pointer to a logical port ID as parameters. It sets the memory	55	19 - DCODE_FEDMODE_DIR 20 - DCODE_FEDMODE_FLYSHT 21 - DCODE_FEDMODE_FLYLNG  Return Value Type Range Description iError short 1 Error flag 1 iError = 0 for success. Nonzero is an error number (see KamMissGetErrorMsg).  KamDecoderGetModelFacility takes a decoder object ID and
1 Opaque object ID handle returned by KamDecoderPutAdd. 2 Maximum value for this server given by KamPortGetMaxLogPorts. Return Value Type Range Description iError short 1 Error flag 1 iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg). KamDecoderMovePort takes a decoder object ID and pointer to a logical port ID as parameters. It sets the memory pointed to by piLogicalPortID to the logical port ID	55	19 - DCODE_FEDMODE_DIR 20 - DCODE_FEDMODE_FLYSHT 21 - DCODE_FEDMODE_FLYLNG  Return Value Type Range Description iError short 1 Error flag 1 iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg).  KamDecoderGetModelFacility takes a decoder object ID and pointer to a decoder facility mask as parameters. It
1 Opaque object ID handle returned by KamDecoderPutAdd. 2 Maximum value for this server given by KamPortGetMaxLogPorts. Return Value Type Range Description iError short 1 Error flag 1 iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg). KamDecoderMovePort takes a decoder object ID and pointer to a logical port ID as parameters. It sets the memory pointed to by piLogicalPortID to the logical port ID associated with IDecoderObjectID.	33	19 - DCODE_FEDMODE_DIR 20 - DCODE_FEDMODE_FLYSHT 21 - DCODE_FEDMODE_FLYLNG  Return Value Type Range Description iError short 1 Error flag 1 iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg).  KamDecoderGetModelFacility takes a decoder object ID and pointer to a decoder facility mask as parameters. It sets the memory pointed to by pdwFacility to the decoder
1 Opaque object ID handle returned by KamDecoderPutAdd. 2 Maximum value for this server given by KamPortGetMaxLogPorts. Return Value Type Range Description iError short 1 Error flag 1 iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg). KamDecoderMovePort takes a decoder object ID and pointer to a logical port ID as parameters. It sets the memory pointed to by piLogicalPortID to the logical port ID associated with IDecoderObjectID.  UKamDecoderCheckAddrinUse	55	19 - DCODE_FEDMODE_DIR 20 - DCODE_FEDMODE_FLYSHT 21 - DCODE_FEDMODE_FLYLNG  Return Value Type Range Description iError short 1 Error flag 1 iError = 0 for success. Nonzero is an error number (see KamMissGetErrorMsg).  KamDecoderGetModelFacility takes a decoder object ID and pointer to a decoder facility mask as parameters. It sets the memory pointed to by pdwFacility to the decoder facility mask associated with iDCCAddr.
1 Opaque object ID handle returned by KamDecoderPutAdd. 2 Maximum value for this server given by KamPortGetMaxLogPorts. Return Value Type Range Description iError short 1 Error flag 1 iError = 0 for success. Nonzero is an error number (see KamMissGetErrorMsg). KamDecoderMovePort takes a decoder object ID and pointer to a logical port ID as parameters. It sets the memory pointed to by piLogicalPortID to the logical port ID associated with IDecoderObjectID.  UKamDecoderCheckAddrInUse Parameter List Type Range Direction Description	33	19 - DCODE_FEDMODE_DIR 20 - DCODE_FEDMODE_FLYSHT 21 - DCODE_FEDMODE_FLYLNG  Return Value Type Range Description iError short 1 Error flag 1 iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg).  KamDecoderGetModelFacility takes a decoder object ID and pointer to a decoder facility mask as parameters. It sets the memory pointed to by pdwFacility to the decoder facility mask associated with iDCCAddr.  0KamDecoderGetObjCount
1 Opaque object ID handle returned by KamDecoderPutAdd. 2 Maximum value for this server given by KamPortGetMaxLogPorts. Return Value Type Range Description iError short 1 Error flag 1 iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg). KamDecoderMovePort takes a decoder object ID and pointer to a logical port ID as parameters. It sets the memory pointed to by piLogicalPortID to the logical port ID associated with IDecoderObjectID. UKamDecoderCheckAddrInUse Parameter List Type Range Direction Description iDecoderAddress int 1 In Decoder address	33	19 - DCODE_FEDMODE_DIR 20 - DCODE_FEDMODE_FLYSHT 21 - DCODE_FEDMODE_FLYSHT 21 - DCODE_FEDMODE_FLYING  Return Value Type Range Description iError short 1 Error flag 1 iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg).  KamDecoderGetModelFacility takes a decoder object ID and pointer to a decoder facility mask as parameters. It sets the memory pointed to by pdwFacility to the decoder facility mask associated with iDCCAddr.  0KamDecoderGetObjCount Parameter List Type Range Direction Description
1 Opaque object ID handle returned by KamDecoderPutAdd. 2 Maximum value for this server given by KamPortGetMaxLogPorts. Return Value Type Range Description iError short 1 Error flag 1 iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg). KamDecoderMovePort takes a decoder object ID and pointer to a logical port ID as parameters. It sets the memory pointed to by piLogicalPortID to the logical port ID associated with IDecoderObjectID.  UKamDecoderCheckAddrInUse Parameter List Type Range Direction Description iDecoderAddress int 1 In Decoder address iLogicalPortID int 2 In Logical Port ID	33	19 - DCODE_FEDMODE_DIR 20 - DCODE_FEDMODE_FLYSHT 21 - DCODE_FEDMODE_FLYSHT 21 - DCODE_FEDMODE_FLYLNG  Return Value Type Range Description iError short 1 Error flag 1 iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg).  KamDecoderGetModelFacility takes a decoder object ID and pointer to a decoder facility mask as parameters. It sets the memory pointed to by pdwPacility to the decoder facility mask associated with iDCCAddr.  0KamDecoderGetObjCount Parameter List Type Range Direction Description iDecoderClass int 1 In Class of decoder
1 Opaque object ID handle returned by KamDecoderPutAdd. 2 Maximum value for this server given by KamPortGetMaxLogPorts. Return Value Type Range Description iError short 1 Error flag 1 iError = 0 for success. Nonzero is an error number (see KamMissGetErrorMsg). KamDecoderMovePort takes a decoder object ID and pointer to a logical port ID as parameters. It sets the memory pointed to by piLogicalPortID to the logical port ID associated with IDecoderObjectID.  UKamDecoderCheckAddrInUse  Parameter List Type Range Direction iDecoderAddress int 1 In Decoder address iDecoderAddress int 2 In Logical Port ID iDecoderClass int 3 In Class of decoder	33	19 - DCODE_FEDMODE_DIR 20 - DCODE_FEDMODE_FLYSHT 21 - DCODE_FEDMODE_FLYLNG  Return Value Type Range Description iError short 1 Error flag 1 iError = 0 for success. Nonzero is an error number (see KamMissGetErrorMsg).  KamDecoderGetModelFacility takes a decoder object ID and pointer to a decoder facility mask as parameters. It sets the memory pointed to by pdwFacility to the decoder facility mask associated with iDCCAddr.  0KamDecoderGetObjCount Parameter List Type Range Direction Description iDecoderClass int 1 In Class of decoder piObjCount int * 0-65535 Out Count of active
1 Opaque object ID handle returned by KamDecoderPutAdd. 2 Maximum value for this server given by KamPortGetMaxLogPorts. Return Value Type Range Description iError short 1 Error flag 1 iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg). KamDecoderMovePort takes a decoder object ID and pointer to a logical port ID as parameters. It sets the memory pointed to by piLogicalPortID to the logical port ID associated with IDecoderObjectID.  UKamDecoderCheckAddrInUse Parameter List Type Range Direction Description iDecoderAddress int 1 In Decoder address iLogicalPortID int 2 In Logical Port ID	60	19 - DCODE_FEDMODE_DIR 20 - DCODE_FEDMODE_FLYSHT 21 - DCODE_FEDMODE_FLYSHT 21 - DCODE_FEDMODE_FLYSHT 21 - DCODE_FEDMODE_FLYSHT  1
1 Opaque object ID handle returned by KamDecoderPutAdd. 2 Maximum value for this server given by KamPortGetMaxLogPorts. Return Value Type Range Description iError short 1 Error flag 1 iError = 0 for success. Nonzero is an error number (see KamMissGetErrorMsg). KamDecoderMovePort takes a decoder object ID and pointer to a logical port ID as parameters. It sets the memory pointed to by piLogicalPortID to the logical port ID associated with IDecoderObjectID.  UKamDecoderCheckAddrInUse  Parameter List Type Range Direction iDecoderAddress int 1 In Decoder address iDecoderAddress int 2 In Logical Port ID iDecoderClass int 3 In Class of decoder	33	19 - DCODE_FEDMODE_DIR 20 - DCODE_FEDMODE_FLYSHT 21 - DCODE_FEDMODE_FLYLNG  Return Value Type Range Description iError short 1 Error flag 1 iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg).  KamDecoderGetModelFacility takes a decoder object ID and pointer to a decoder facility mask as parameters. It sets the memory pointed to by pdwFacility to the decoder facility mask associated with iDCCAddr.  OKamDecoderGetObjCount Parameter List Type Range Direction Description iDecoderClass int 1 In Class of decoder piObjCount int * 0-65535 Out Count of active decoders

23

24

-continued	-continued
	***** · · · · · · · · · · · · · · · · ·
LICATION PROGRAMMING INTERFACE	APPLICATION PROGRAMM

3 - DECODER_SENSOR_TYPE.  Return Value Type Range Description*  If ill ror of 0 for success. Nonzero is an error number (see KamiMicoGetTrorMsq).  RomDecoderGetGOffCollocant to the count of active decoder of the type given by iDecoderCollocant to the count of active decoders of the type given by iDecoderCollocal folder.  Parameter List Type Range Direction Description* Indeex in 1 In Decoder army index IDecoderClass in 2 In Class of decoder object ID and Decoder army index ID 10 to (KamDecoderGetAMdreasCount -1)  O 10 to (KamDecoderGetTrorMsq).  SamDecoderGetGetTrorMsq).  RambecoderGetGoffCollocant to the count of active decoder object ID and Decoder and the country of the country o	
Return Value Type Range Description- Effort short 1 Error flag 1 litror = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg). KamDecoderGetGetDicCount takes a decoder class and a pointer to an address ocut as parameters. It is test the memory pointed to by picolic closes.  OKamDecoderGetGetDicCount takes a decoder class and a pointer to an address ocut as parameters. It is to the count of active decoders of the type given by DiecoderClass.  OKamDecoderGetGetColjcAlIndex Parameter List Type Range Direction Description index int 1 In Decoder army index pilocetal Dison and the property of the property	APPLICATION PROGRAMMING INTERFACE
Return Value Type Range poscription* Eitror short 1 Error flag 1 Eitror a - O for success. Nonzero is an error number (see KamMiscoeffernovMag). KamDecoderGetObjCount takes a decoder class and a pointer to an address count at sparameter. It is test the memory pointed to by pjObjCount to the count of active decoders of the type given by DecoderClass.  OKamDecoderGetObjCoUnity Takes a Direction Description Decoder in 1 In Decoder array indicate piDecoderClobjectID only 2 - Out Decoder array indicate piDecoderClobjectID only 3 - Out Decoder array indicate piDecoderClobjectID only 3 - Out Decoder array indicate piDecoderClobjectID only 4 - Out array indicate piDecoderClobjectID only 5 - Out Decoder array indicate piDecoderClobjectID only 5 - Out 1 - Ou	5 Parameter List Type Range Direction Description
iError 1 for success. Nonzero is an error number (see KamMiscGetErrorMsg.).  KamDecodefciolOcount takes a decoder class and a pointer to an address count as a parameters. It sets the memory pointed to by piObjCount to the count of active decoders of the type given by iDecoderClass.  KamDecodefciolOcount Count of active decoders of the type given by iDecoderClass.  Range Direction Description in the count of active decoders of the type given by iDecoderClass.  Range Direction Description in the count of active decoders of the type given by iDecoderClass.  Range Direction Description in the count of active decoder of piDecoderClass.  The count of the count of active decoder of piDecoderObjectID in grange Direction Description in the count of active decoder of piDecoderObjectID in grange Direction Description in the count of active decoders of the count of the count of active decoders of the count of the count of the count of active decoders of the count of the count of the count of active decoders of the count of active decoders of the count of the count of the count of active decoders of the count of the co	
1    Opaque object ID handle returned by KamDecoderFollOpCount takes a decoder class and a pointer to nan address count as parameters. It sets the memory pointed to by piOpCoderClass.  Office type given by iDecoderClass.  Office type given by idecoder given given by idecoder.  Office type given by idea given g	
KamDecoderFold-Discount takes a decoder class and a pointer to an address count a sparameters. It sets the memory pointed to by piDecoderClass and to by piDecoderClass and to the type given by DecoderClass and to the property of the p	manufacturer name
ion an address count as parameters. It sets the memory pointed to by picib/Count to the count of active decoders of the type given by iDecoderClass.  Index in 1	
pointed to by piObjCount to the count of active decoders of the type given by jDecoderClass.  OKAnDecoderGetObjAtIndex Parameter List Type Range Direction Description* DiPocoderClas in 1 2 In Class of decoder array index int 1 1 In Decoder array index int 1 2 In Class of decoder object ID and pointer to decoder object ID 1 0 to (KamDecoderGetAddressCount - 1).  1 0 to (KamDecoderGetAddressCount - 1). 2 1 DECODER_ENGINE_TYPE, 2 DECODER_ENGINE_TYPE, 3 DECODER_SENGINE_TYPE, 4 DECODER_SENGINE_TYPE, 4 DECODER_SENGINE_TYPE, 5 DECODER_	
of the type given by iDecoderClass.  Parameter List Type Range Direction Description index in 1 in Decoder array index in DecoderClass int 2 in Class of decoder polipect ID on the Direction Color of the Decoder Class int 2 in Decoder array index object ID and polipecoderClass int 2 in Class of decoder object ID and polipecoderClass place (See Earn) Miscoccide (See Earn) M	
Items	
Parameter List Type Range Direction Description illocoder(Class in 1 2 In Class of decoder object ID and politer to the coder object ID and politer to the coder object ID and politer to the coder object ID and politer to a manufacturer and straining as parameters. It is the memory polited to by pbsMfgName to the name: the decoder model plant the same object ID and politer to a manufacturer and straining as parameters. It is stitle the decoder model politer to a manufacturer and straining as parameters. It is stitle memory polited to by pbsMfgName to the name: the decoder model politer to a manufacturer and straining as parameters. It is stitle memory polited to by pbsMfgName to the name: the decoder model placeder(Dipcount takes a decoder index, decoder (see KamMissGetErrorMsg).  KamDecoder(Dipcount takes a decoder index, decoder class, and a positer to an object ID as parameters. It is estet the memory pointed to by ploecoder(Dipcount takes a decoder index, decoder class, and a positer to an object ID as parameters. It is estent memory pointed to by ploecoder(Dipcount takes a decoder index, decoder class, and a positer to an object ID and port ID object ID	
Simple   Decoder   Time   Decoder army index   Decoder   DipecoderChipectID   Ong   * 3	
iDecoderClass int 2 In Class of decoder polipectoderObjectID long * 3 Out Pointer to decoder annual facture rame as string as parameters. It sets the memory pointed to by pbsMfgName to the name of the decoder manufacturer annual manufacturer annual facturers. It sets the memory pointed to by pbsMfgName to the name of the decoder manufacturer. The manufacturer has been as the decoder manufacturer annual facturers. It sets the memory pointed to by pbsMfgName to the name of the decoder manufacturer. The manufacturer has been decoder manufacturer. The manufacturer has been decoder manufacturer. The manufacturer has been decoder manufacturer annual facturers. The sets the memory pointed to by piDecoderObjectID to a parameters. It is the decoder manufacturer annual facturer. The decoder decoder object ID as parameters. It is the decoder manufacturer annual facturers. The decoder object ID as parameters. It is the decoder manufacturer annual facturers. The decoder object ID as parameters. It is the decoder manufacturer annual facturer. The parameter is the decoder manufacturer. The manufacturer annual facturer. The decoder object ID as parameters. It is estimated the decoder manufacturer annual facturer. The following the decoder manufacturer annual facturer. The following the decoder manufacturer annual facturer. The following the decoder manufacturer. The following the following the foll	
plDecoderObjectID long * 3 Out Pointer to decoder object ID 1 to (KamDecoderGetAddressCount - 1).  1 0 to (KamDecoderGetAddressCount - 1). 2 1 - DECODER_RENGINE_TYPE, 3 - DECODER_TENDINE_TYPE, 3 - DEC	The second secon
object ID  1 0 to (KamDecoderGetAddressCount - 1). 2 1 · DECODER_ SNTOKE_TYPE, 2 · DECODER_ SENSOR_TYPE, 3 · DECODER_ SENSOR_TYPE, 3 · DECODER_ SENSOR_TYPE, 3 · DECODER_ SENSOR_TYPE, 4 Command to the name in decoder manufacture. 5 Opaque object ID handle returned by  KamDecoderGetAddres 1	
1 0 to (KamDecoderGetAddressCount - 1). 2 1 - DECODER_ SWITCH_TYPE, 3 - DECODER_SWITCH_TYPE, 4 - DECODER_TWITCH_TYPE, 4 - DECODER_SWITCH_TYPE, 4 - DECODER_SWITCH_TYPE, 4	
2 - DECODER_SWITCH_TYPE. 3 - Opaque object ID handle returned by  KamDecoderPutAdd. Return Value Type Range Description Elfror short 1 Error flag 1 Eiror of for success. Nonzero is an error number (see KamMiscGetErrorMsg). KamDecoderGetObjecut lakes a decoder index, decoder class, and a pointer to an object ID aparameters. It sests the memory pointed to by pIDecoderObjectID to the selected object ID.  OKamDecoderPutAdd Parameter List Type Range Direction Description DecoderAddress int 1 In Decoder address il.ogicalCmdPortID int 1-65535 2 In Logical command port ID iclogicalCmdPortID int 1-65535 2 In Logical command port ID iclogicalProgPortID int 1-65535 2 In Logical command port ID iclogicalProgPortID int 1 - Opaceder address int 1 In Decoder model type ID piDecoderObjectID long * 5 Out Decoder object ID and Decoder object ID as parameters. It ereates a new locomotive object ID used by the server as a key.  OkamDecoderPutAdd takes a decoder object ID as parameters. It creates a new locomotive object ID used by the server as a key.  OkamDecoderPutAdd takes a decoder object ID as parameters. It creates a new locomotive object ID used by the server as a key.  OkamDecoderPutAdd takes a decoder object ID as parameters. It creates a new locomotive object ID used by the server as a key.  OkamDecoderPutAdd takes a decoder object ID as parameters. It creates a new locomotive object ID used by the server as a key.  OkamDecoderPutObjectID long 1 In Decoder object ID as parameters. It creates a new locomotive object ID used by the server as a key.  OkamDecoderPutObjectID long 1 In Decoder object ID and Decoder object ID and Decoder object ID as parameters. It creates a new locomotive object in the locomotive database and sets the memory pointed to by piSpeedStep to the server copy of the engine variables such speed obj	
3 - DECODER, SENSOR_TYPE. 3 Opaque object ID handle returned by  KamDecoderPutAdd.  Return Value Type Range Description  Elfror 1 flag  1 Elfror - 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg).  KamDecoderGetObjCount takes a decoder index, decoder class, and a pointer to an object ID as parameters. It sets the memory pointed to by pIDecoderObjectID to the selected object ID.  OKamDecoderPutAdd  Parameter 1 List Type Range Direction Description iDecoderAddress int 1 In Decoder address int 1 Logical command command port ID it iLogicalProgPortID int 1 -65535 2 In Logical command port ID it iClearState int 3 In Clear state flag iModel int 4 In Decoder model type ID pIDecoderObjectID long be to the server given by KamDecoderGetMaxModels.  3 0 - retain state, 1 - clear state.  2 Maximum value for this server given by KamDecoderCetMaxModels.  3 0 - retain state, 1 - clear state.  4 Maximum value for this server given by KamDecoderCetMaxModels.  5 Opaque object ID handle. The object ID is used to reference the decoder.  5 Opaque object ID handle. The object ID is used to reference the decoder model ID, and a pointer to a decoder object ID as parameters. It is used to reference the decoder model ID, and a pointer to a decoder object ID to sure as a key.  OkamDecoderPutAdd takes a decoder object ID to command logical port, clear flag in it creates a new to comotive object ID to sure the command logical port, programming logical port, clear flag in iteration to be programming policition to the server as a key.  OkamDecoderPutAdd takes a decoder object ID to command logical port, clear flag in iteration to be programming policition to the server as a key.  OkamDecoderPutAdd takes a decoder object ID to command logical port, clear flag in iteration to be programming policition to the server as a key.  OkamDecoderPutAdd takes a decoder object ID to command logical port, clear flag in iteration to be programming policition to the maximum supported speed step as parameters. It sets the memory be added to t	0KamDecoderGetPowerMode
AmDecoderPutAdd.  Return Value   Type   Range   Description	
Agange object ID handle returned by Rampe Description in itror shot 1 Error flag 1 iitror = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg).  KamDecoderGetChicoCount takes a decoder index, decoder class, and a pointer to an object ID and pointer to the selected object ID.  SiLogicalCmdPortID int 1 -65535 2 In Logical command port ID in Decoder address in 1 In Decoder model type ID placederObjectID long * 5 Out Decoder by the server given by KamPortGetMaxIogPorts.  1 1-127 for short locomotive addresses 1-10239 for long locomotive decoders. 0-511 for accessory decoders. 3 0 - retain state, 1 - clear state. 4 Maximum value for this server given by KamPocoderGetMaxIogPorts. 3 0 - retain state, 1 - clear state. 4 Maximum value for this server given by KamPocoderGetMaxIogPorts. 5 Opaque object ID handle. The object ID is used to reference the decoder. Cean Maximum value for this server given by KamDecoderGetMaxIogPorts. 5 Opaque object ID handle. The object ID is used to reference the decoder object ID sand by the decoder object ID san	
Return Value   Type   Range   Description	posrowermode BSTR 2 Out Pointer to
Elrror short   1   Error flag   2   Error flag   Error flag   2   Error flag   2   Error flag   2   Error flag   Error flag   2   Error flag   2   Error flag   2   Error flag   Error flag   2   Error flag   2   Error flag   Error flag   Error	decoder power
1 iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg).  KamDecoderGetObjCount takes a decoder index, decoder class, and a pointer to an object ID as parameters. It sets the memory pointed to by piDecoderObjectID to the selected object ID.  CKamDecoderPutAdd  Parameter List	
(see KamMiscGetErrorMsg).  2 Exact return type depends on language. It is sets the memory pointed to by pIDecoderObjectID to the selected object ID.  30 Farameter List	
KamDecoderGet(Di)Count lakes a decoder index, decoder class, and a pointer to an object ID as parameters. It class the memory pointed to by piDecoderObjectID to the selected object ID.    Command	
class, and a pointer to an object ID as parameters. It sets the memory pointed to by piDecoderObjectID to the selected object ID.  OKamDecoderPutAdd Parameter List Type Range Direction Description iDecoderAddress int 1 In Decoder address iI. OgicalCmdPortID int 1-65535 2 In Logical command port ID ii. OgicalProgPortID int 1-65535 2 In Logical programming port ID ii. OgicalProgPortID int 1-65535 2 In Logical programming port ID ii. OgicalProgPortID int 1-65535 2 In Logical programming port ID ii. OgicalProgPortID int 1-65535 2 In Logical programming port ID ii. OgicalProgPortID int 1-65535 2 In Logical programming port ID ii. OgicalProgPortID int 1-65535 2 In Logical programming port ID ii. OgicalProgPortID int 1-65535 2 In Logical programming port ID ii. Opecader model type ID IDecoderObjectID long * 5 Out Decoder object ID opecader object ID is used to reference the decoders. O-511 for accessory decoders. Object ID sused to reference the decoder.  2 Maximum value for this server given by KamDecoderGetMaxModels.  3 O - retain state, 1 - clear state. A Maximum value for this server given by KamDecoderGetMaxModels.  5 Opaque object ID handle. The object ID is used to reference the decoder. Return Value Type Range Description iError short 1 Error flag I iIrror = 0 for success. Nonzero is an error number (see KamMiseGetErrorNsg).  KamDecoderGetMaxModels.  5 Opaque object ID handle returned by KamDecoderSutAdd takes a decoder object ID is used to reference the decoder.  Return Value Type Range Description iError short 1 Error flag I iIrror = 0 for success. Nonzero is an error number (see KamMiseGetErrorNsg).  KamDecoderGetMaxModels.  5 Opaque object ID handle returned by KamDecoderSutAdd takes a decoder object ID is used to reference the decoder.  Return Value Type Range Direction Description iError short 1 Error flag I iIrror = 0 for success. Nonzero is an error number (see KamMiseGetErrorNsg).  KamDecoderGetMaxModels.  5 Opaque object ID handle returned by programming logical port, programming logical port, progra	
sets the memory pointed to by pIDecoderObjectID to the selected object ID.    SamDecoderHuAddress   Type   Range   Direction   Description   Description   Decoder Address   Direction   Description   Decoder Address   Direction   Description   Decoder Address   Direction   Description   Decoder Address   Direction   Decoder Address   Direction   Description   Decoder Address   Direction   Decoder   Directi	
selected object ID.  OKamDecoderPutAdd  Parameter List Type Range Direction Description iDecoderAddress int 1 In Decoder address int 2 In Decoder mand port ID In Decoder mand port ID In Decoder model type ID piDecoderObjectID long * 5 Out Decoder object ID addresses. 1 - 10239 for long locomotive decoders. 0 - 511 for accessory decoders. 2 Maximum value for this server given by KamPerofertMaxLogPorts. 3 O - retain state, 1 - clear state.  4 Maximum value for this server given by KamPerofertMaxLogPorts. 3 O - retain state, 1 - clear state.  4 Maximum value for this server given by KamDecoderGetMaxModels. 5 Opaque object ID handle. The object ID is used to reference the decoder. Return Value Type Range Description iError short 1 Error flag (see KamMiscGetErrorMsg). KamDecoderGetMaxModels. 5 Opaque object ID handle. The object ID is used to reference the decoder. Return Value Type Range Description iError short 1 Error flag. A Commands to control locomotive decoders. Near-point of the server is an error number (see KamMiscGetErrorMsg). KamDecoderGetMaxModels. 5 Opaque object ID handle returned by KamDecoderGetMaxModels. 5 Opaque object ID and becoder object ID and becoder object ID and becoder ob	
OKamDeoderPutAdd Parameter List Type Range Direction Description D	· · · · · · · · · · · · · · · · · · ·
Parameter List   Type   Range   Direction   Decoder address   In	
In   Decoder address	
iLogicalProgPortID int 1-65535 2 In Logical command point ID intogralProgPortID int 1-65535 2 In Logical programming port ID int 3 In Clear state flag indeed int 4 In Decoder model type ID plDecoderObjectID long 5 5 Out Decoder model type ID plDecoderObjectID long 6 5 Out Decoder model type ID plDecoderObjectID long 7 5 Out Decoder model type ID plDecoderObjectID long 8 5 Out Decoder model type ID plDecoderObjectID long 8 5 Out Decoder model type ID plDecoderObjectID long 8 5 Out Decoder model type ID plDecoderObjectID long 8 5 Out Decoder long locomotive decoders. O-511 for accessory decoders. O-511 for accessory decoders. O-512 for long locomotive decoders. O-513 for accessory decoders. O-514 for accessory decoders. O-515 for accessory decoders	
command port ID   1   1   1   1   1   1   1   1   1	
iLogicalProgPortID   int   1-65535 2   In   Logical   programming   port ID   port ID	
programming port ID 10 10 10 10 10 10 10 10 10 10 10 10 10	0KamDecoderGetMaxSpeed
iClearState int 3 In Clear state flag iModel int 4 In Decoder model type ID piDecoderObjectID long 5 Out Decoder object ID long 5 Out Decoder object ID long 6 S Out Decoder object ID long 6 S Out Decoder object ID long 1-127 for short locomotive addresses. 1-10239 for long locomotive decoders. 0-511 for accessory decoders. 100 glocomotive decoders. 0-512 for accessory decoders. 100 glocomotive decoders. 0-512 for accessory decoders. 100 glocomotive decoders. 100	
iClearState int 3 In Clear state flag iModel int 4 In Decoder model type ID plDecoderObjectID long * 5 Out Decoder object ID becoder object ID 1-127 for short locomotive addresses. 1-10239 for long locomotive decoders. 0-511 for accessory decoders. 2 Maximum value for this server given by KamPortGetMaxLogPorts. 3 0 - retain state, 1 - clear state. 4 Maximum value for this server given by KamDecoderGetMaxModels. 5 Opaque object ID handle. The object ID is used to reference the decoder. Return Value Type Range Description iError short 1 Error flag Control in Error flag Description in Error flag DecoderPutAdd takes a decoder object ID, command logical port, programming logical port, clear flag, parameters. It creates a new locomotive object in the locomotive database and sets the memory pointed to by piPlecoderObjectID to the decoder object ID used by the server as a key. 0KamDecoderPutAdd. 2 14, 28, 56, or 128 for locomotive decoders. 0 for accessory decoders. 1 iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg). KamDecoderGetMaxSpeed takes a decoder object ID and pointer to the maximum supported speed step as parameters. It sets the memory pointed to by piSpeedStep to the maximum supported speed step as parameters. It sets the memory pointed to by piSpeedStep to the maximum supported speed step as parameters. It sets the memory pointed to by piSpeedStep to the maximum supported speed step as parameters. It sets the memory pointed to by piSpeedStep to the maximum supported speed step as parameters. It sets the memory pointed to by piSpeedStep to the maximum supported speed step as parameters. It sets the memory pointed to by piSpeedStep to the maximum supported speed step as parameters. It sets the memory pointed to by piSpeedStep to the maximum supported speed step as parameters. It sets the memory pointed to by piSpeedStep to the maximum supported speed step as parameters. It sets the memory pointed to by piSpeedStep to the maximum supported speed step as parameters. It sets the memory	
iModel int 4 In Decoder model type ID piDecoderObjectID long * 5 Out Decoder object ID Decoder object ID long * 5 Out Decoder object ID Decoder object ID long * 5 Out Decoder object ID long in 1–127 for short locomotive addresses. 1–10239 for long locomotive decoders. 0–511 for accessory decoders. 2 Maximum value for this server given by KamPortGetMaxLogPorts. 3 0 - retain state, 1 - clear state. 4 Maximum value for this server given by KamDecoderGetMaxModels. 5 Opaque object ID handle. The object ID is used to reference the decoder. Return Value Type Range Description liError short 1 Error flag parameters. It is the memory pointed to by piSpeedStep (see KamMiscGetErrorMsg). 4 A Commands to control locomotive decoders. Nonzero is an error number (see KamMiscGetErrorMsg). 4 A Commands to control locomotive decoders. Nonzero is an error number (see KamMiscGetErrorMsg). 5 Opaque object ID handle returned by KamDecoderGetMaxAodels. 5 Opaque object ID substitute Type Range Description is increased and set of the maximum supported speed step as parameters. It sets the memory pointed to by piSpeedStep As parameters. It sets the memory pointed to by piSpeedStep This section describes the commands that control locomotive decoders. These commands that control locomotive decoders. These commands that control locomotive decoders. These commands that control locomotive decoders. Opaque object ID handle returned by KamDecoderPutAdd to accessory decoders. 40 It is Error flag 1 it iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg). KamDecoderGetMaxSpeed takes a decoder object ID and pointer to the maximum supported speed step as parameters. It sets the memory pointed to by piSpeedStep This section describes the commands that control locomotive decoders. This section describes the commands control locomotive decoders. These commands control locomotive decoders of things such as locomotive speed and direction. For efficiency, a copy of all the engine variables such speed logical port, clear flag, control	33 1 1 1
plDecoderObjectID long * 5 Out Decoder object ID 1 1-127 for short locomotive addresses. 1-10239 for long locomotive decoders. 0-511 for accessory decoders. 2 Maximum value for this server given by KamPortGetMaxLogPorts. 3 0 - retain state, 1 - clear state. 4 Maximum value for this server given by KamDecoderGetMaxModels. 5 Opaque object ID handle. The object ID is used to reference the decoder. Return Value Type Range Description iterror short 1 Error flag 1 iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg). KamDecoderPutAdd takes a decoder object ID, command logical port, clear flag, decoder model ID, and a pointer to a decoder object ID as parameters. It creates a new locomotive object in the locomotive database and sets the memory pointed to by piSpeeddiency, a copy of all the engine variables such speed locomotive decoders. This section describes the commands that control locomotive decoders.  A Commands to control locomotive decoders. 45 to the maximum supported speed step as parameters. It sets the memory pointed to by piSpeedStep of the maximum speed step supported by the decoder.  A Commands to control locomotive decoders. These commands that control locomotive decoders of the maximum supported speed step as parameters. It sets the memory pointed to by piSpeedStep of the maximum speed step supported by the decoder.  A Commands to control locomotive decoders. These commands that control locomotive decoders. These commands that control locomotive decoders of the maximum supported speed step as parameters. It sets the memory pointed to by piSpeedStep of the maximum supported speed step as parameters. It sets the memory pointed to by piSpeedStep of the maximum supported speed step as parameters. It sets the memory pointed to by piSpeedStep of the maximum supported speed step as parameters. It sets the memory pointed to by piSpeedStep of the maximum supported speed step as parameters. It sets the memory pointed to by piSpeedStep of the maximum supported speed step as parameters. It	speed step
object ID 1 1-127 for short locomotive addresses. 1-10239 for long locomotive decoders. 0-511 for accessory decoders. 2 Maximum value for this server given by KamPortGetMaxLogPorts. 3 0 - retain state, 1 - clear state. 4 Maximum value for this server given by KamDecoderGetMaxModels. 5 Opaque object ID handle. The object ID is used to reference the decoder. Return Value Type Range Description iError short 1 Error flag 1 iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg). KamDecoderPutAdd takes a decoder object ID, command logical port, programming logical port, clear flag, decoder model ID, and a pointer to a decoder object ID as parameters. It creates a new locomotive object in the locomotive database and sets the memory pointed to by plDecoderObjectID to the decoder object ID used by the server as a key.  0KamDecoderPutDel Range Direction Description long 1 In Decoder object ID locoderObjectID long 1 In Decoder ObjectID long 1 In Decoder ObjectID long 1 In Decoder ObjectID long 1 In Decoder object ID long 1 In Decoder ID, some accessory decoders.  2 14, 28, 56, or 128 for locomotive decoders. 0 for accessory decoders.  440 IError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg).  KamDecoderGetMaxModels.  5 Opaque object ID handle. The object ID is used to reference the decoder.  Return Value Type Range Description in IError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg).  KamDecoderGetMaxModels.  A Commands to control locomotive decoders. This section describes the commands that control locomotive decoders. These commands such as KamEngGetSpectory of all the engine variables. You can send all changes to the engine using the KamCmdCommand command.  0KamEngGetSpeed  55 Parameter List Type Range Direction Description long 1 In DecoderObjectID	
1 1-127 for short locomotive addresses. 1-10239 for long locomotive decoders. 0-511 for accessory decoders. 2 Maximum value for this server given by KamPortGetMaxLogPorts. 3 0 - retain state, 1 - clear state. 4 Maximum value for this server given by KamDecoderGetMaxModels. 5 Opaque object ID handle. The object ID is used to reference the decoder. 8 Return Value Type Range Description iError short 1 Error flag Sieror short 1 Sieror = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg). 4 Commands to control locomotive decoders. This section describes the commands that control locomotive decoders. This section describes the commands that control locomotive decoders. These commands control locomotive speed and direction. For efficiency, a copy of all the engine variables such speed logical port, clear flag, control locomotive decoders. These commands control locomotive decoders. These commands such as KamEngGetSpeed step sa parameters. It creates a new locomotive object ID so decoder object ID as parameters. It creates a new locomotive object in the locomotive decoders. These commands such as KamEngGetSpeed step sa parameters. It creates a new locomotive object ID so decoder object ID as comminicated only with the server, not the actual decoder. You should first make any changes to the engine variables. You can send all changes to the engine variables. You can send all changes to the engine variables. You can send all changes to the engine variables. You can send all changes to the engine variables. You can send all changes to the engine variables. You can send all changes to the engine variables. You can send all changes to the engine variables. You can send all changes to the engine variable	
Return Value   Type   Range   Description	
2 Maximum value for this server given by KamDetGetMaxLogPorts. 3 0 - retain state, 1 - clear state. 4 Maximum value for this server given by KamDecoderGetMaxModels. 5 Opaque object ID handle. The object ID is used to reference the decoder. Return Value Type Range Description iError short 1 Error flag 1 iError = 0 for success. Nonzero is an error number (see KamMissGetErrorMsg). KamDecoderPutAdd takes a decoder object ID, command logical port, clear flag, decoder model ID, and a pointer to a decoder object ID as parameters. It creates a new locomotive object in the locomotive database and sets the memory pointed to by pleased the server as a key.  OKamDecoderPutDel Range Direction Description locomotive decoders.  A. Commands to control locomotive decoders. These commands that control locomotive speed and direction. For efficiency, a copy of all the engine variables such as KamEngGetSpeed (oKamDecoderPutDel Range Direction Description) long 1 In Decoder object ID long 2 In Decoder object ID long 2 In Decoder object ID long 3 In Decoder object ID long 4 In Decoder object ID long 4 In Decoder object ID long	
This section describes the commands that control locomotive decoders. Nonzero is an error number of start of the server, not the actual decoder object ID as parameters. It creates a new locomotive object ID used to by pilocoderObjectID to the decoder object ID used to by pilocoderObjectID to the decoder object ID used to the maximum supported speed step as parameters. It sets the memory pointed to by pilocoder object ID and pointer to the maximum supported speed step as parameters. It sets the memory pointed to by pilocoder object ID to the decoder.    A   Commands to control locomotive decoders	
3 0 - retain state, 1 - clear state.  4 Maximum value for this server given by  KamDecoderGetMaxModels.  5 Opaque object ID handle. The object ID is used to reference the decoder.  Return Value Type Range Description iError short 1 Error flag  1 iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg).  KamDecoderPutAdd takes a decoder object ID, command logical port, clear flag, decoder model ID, and a pointer to a decoder object ID as parameters. It creates a new locomotive object in the piDecoderObjectID to the decoder object ID used by the server as a key.  OKamDecoderPutDel Server as a key.  OKamDecoderPutDel Server as a key.  OKamDecoderPotDel Type Range Direction Description liDecoderObjectID long 1 In Decoder ObjectID long 1 In Decoder object ID long 1 In Decoder ObjectID long 2 Out Pointer to the maximum supported speed step as parameters. It sets the memory pointed to by piSpeedStep object Exparameters. It sets the memory pointed to by piSpeedStep as parameters. It sets the memory pointed to by piSpeedStep as parameters. It sets the memory pointed by the decoder.  A. Commands to control locomotive decoders. These commands that control locomotive decoders. These commands control things such as locomotive speed and direction. For efficiency, a copy of all the engine variables such speed locomotive object ID and pointer to the maximum supported by the decoder.  A. Commands to control locomotive decoders. These commands that control locomotive decoders. These commands that control locomotive speed and direction. For efficiency, a copy of all the engine variables such speed locomotive object ID and springers in the server. Commands such as KamEngGetSpeed step supported by the decoder.  A. Commands to control locomotive decoders. These commands such	
A Maximum value for this server given by KamDecoderGetMaxModels.  S Opaque object ID handle. The object ID is used to reference the decoder.  Return Value Type Range Description iError short 1 Error flag  1 iError = 0 for success. Nonzero is an error number (see KamMissGetErrorMsg).  KamDecoderPutAdd takes a decoder object ID, command logical port, clear flag, decoder model ID, and a pointer to a decoder object ID as parameters. It creates a new locomotive object in the locomotive database and sets the memory pointed to by piSpeedStep supported by the decoder. A. Commands to control locomotive decoders. These commands that control locomotive decoders. These commands that control locomotive decoders. These commands that control locomotive speed and direction. For efficiency, a copy of all the engine variables such speed locomotive database and sets the memory pointed to by piSpeedStep to the maximum supported to by the decoder.  A. Commands to control locomotive decoders. These commands control things such as locomotive speed and direction. For efficiency, a copy of all the engine variables such speed communicate only with the server, now the actual decoder. You should first make any changes to the engine variables. You can send all changes to the engine variables. You can send all changes to the engine using the KamCmdCommand to control locomotive decoders.  Communicate only with the server, now the actual decoder. You should first make any changes to the engine variables. You can send all changes to the engine using the KamCmdCommand to control locomotive decoders. These commands that control locomotive decoders. These commands that control locomotive decoders. These commands that control locomotive decoders. These commands were deficiency, a copy of all the engine variables. You can send all changes to the engine variables. You can send all changes to the engine variables. You can send all changes to the engine variables. You can send all changes to the engine variables. You can send all changes to the engin	
KamDecoderGetMaxModels.  Opaque object ID handle. The object ID is used to reference the decoder.  Return Value Type Range Description iError short 1 Error flag  1 iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg.)  KamDecoderPutAdd takes a decoder object ID, command logical port, programming logical port, clear flag, decoder model ID, and a pointer to a decoder object ID as parameters. It creates a new locomotive object in the server as a key.  OKamDecoderPutDel reserver as a key.  OKamDecoderPutDel reserver as a key.  OKamDecoderObjectID long 1 In In Decoder object ID  Description in the maximum supported speed step as parameters. It sets the memory pointed to by piSpeedStep as the parameters. It sets the memory pointed to by piSpeedStep as the parameters. It sets the memory pointed to by piSpeedStep as the maximum supported speed step as the parameters. It sets the memory pointed to by piSpeedStep as the memory pointed to be parameters. It sets the memory pointed to be the maximum supported by the decoder.  A. Commands to control locomotive decoders. These commands control locomotive decoders. These comm	
S Opaque object ID handle. The object ID is used to reference the decoder.  Return Value Type Range Description i Error short 1 Error flag  1 iError = 0 for success. Nonzero is an error number (see KamMissGetErrorMsg).  KamDecoderPutAdd takes a decoder object ID, command logical port, programming logical port, clear flag, decoder model ID, and a pointer to a decoder object ID as parameters. It creates a new locomotive object in the piDecoderObjectID to the decoder object ID used by the server as a key.  OKamDecoderPutDel Parameter List Type Range Direction Description logo 1 In In Decoder object ID logo 1 In In Decoder object ID logo 1 In In Decoder object ID logo Interval ID logo 2 Interval Interval Interval ID logo 2 Interval Interval Interval Interval Interval ID logo 2 Interval Interval Interval ID logo 2 Interval Interval ID logo 2 Interval Interval ID logo 2 Interval I	
reference the decoder.  Return Value Type Range Description  Error flag  1 Error = 0 for success. Nonzero is an error number (see KamMiscGetEtrorMsg).  KamDecoderPutAdd takes a decoder object ID, command logical port, programming logical port, clear flag, decoder model ID, and a pointer to a decoder object ID as parameters. It creates a new locomotive object in the plDecoderObjectID to the decoder object ID used by the server as a key.  OKamDecoderPutDel  Parameter List Type Range Direction Description lDecoderObjectID long 1 In Decoder object ID  Server as a key.  45 to the maximum speed step supported by the decoder: A Commands to control locomotive decoders. These commands tontrol things such as locomotive speed and direction. For efficiency, a copy of all the engine variables such speed locomotive database and sets the memory pointed to by the engine variables. You can send all changes to the engine using the KamCmdCommand command.  OKamEngGetSpeed  OKamEngGetSpeed  OKamEngGetSpeed  Type Range Direction Description lDecoderObjectID long 1 In Decoder ObjectID long 2 Out Pointer Obj	
Return Value Type Range Description  I Error flag  This section describes the commands that the control locomotive decoders. These commands that the control locomotive decoders. The section describes the commands that the control locomotive decoders. The section describes the commands that the control locomotive decoders. The section describes the commands that the control locomotive decoders. The section describes the commands that the control locomotive decoders. The section describes the commands that the control locomotive decoders. The section describes the commands that the control locomotive decoders. The section describes the commands that the control locomotive decoders. The section describes the commands that the control locomotive decoders. The section describes the commands that the control locomotive decoders. The section describes the commands that the control locomotive decoders. The section describes the commands that the control locomotive speed and direction. For efficiency, a copy of all the engine variables such as locomotive speed and direction. For efficiency, a copy of all the engine variables such as locomotive speed and direction. For efficiency, a copy of all the engine variables such as locomotive speed and direction. For efficiency, a copy of all the engine variables such as locomotive speed and direction. For efficiency, a copy of all the engine variables such as locomotive speed and direction. For efficiency, a copy of all the engine variables. You should first make any changes to the engine variables. You can send all changes to the engine using the Kam'CmdCorror command.  OKAMENGETER TOPM SETTING THE PROPRIES TOPM SETTING	
iError short 1 Error flag  This section describes the commands that 1 iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg).  KamDecoderPutAdd takes a decoder object ID, command logical port, programming logical port, clear flag, decoder model ID, and a pointer to a decoder object ID as parameters. It creates a new locomotive object in the locomotive database and sets the memory pointed to by pIDecoderObjectID to the decoder object ID used by the server as a key.  OKamDecoderPutDel  Parameter List Type Range Direction Description locoderObjectID long 1 In Decoder object ID  This section describes the commands that control locomotive decoders. These commands that control locomotive decoders. These commands control locomotive decoders. These commands control locomotive decoders. These commands that control locomotive decoders. These commands control locomotive decoders. These commands that control locomotive decoders. These locomotive decoders. These deficiency, a copy of all the engine variables such speed to stored in the server. Commands such as KamEngGetSpeed you should first make any changes t	
1 iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg).  KamDecoderPutAdd takes a decoder object ID, command logical port, programming logical port, clear flag, decoder model ID, and a pointer to a decoder object ID as parameters. It creates a new locomotive object in the locomotive database and sets the memory pointed to by pIDecoderObjectID to the decoder object ID used by the server as a key.  OKamDecoderPutDel  Parameter List  Type  Range  Direction  Description  Description  DecoderObjectID long  1 In  DecoderObjectID long  Control locomotive decoders. These commands control things such as locomotive speed and direction. For things such as locomotive speed and direction. For efficiency, a copy of all the engine variables such speed of sistored in the server. Commands such as KamEngGetSpeet  You should first make any changes to the server copy of the engine variables. You can send all changes to the engine using the KamCmdCommand command.  OKamEngGetSpeed  OKamEngGetSpeed  Direction  Description  Descri	
KamDecoderPutAdd takes a decoder object ID, command logical port, programming logical port, clear flag, so decoder model ID, and a pointer to a decoder object ID as command a pointer to a decoder object ID as parameters. It creates a new locomotive object in the piDecoderObjectID to the accoder object ID used by the server as a key.  OKamDecoderPutDel Parameter List Type Range Direction Description long of the properties o	
logical port, programming logical port, clear flag, decoder model ID, and a pointer to a decoder object ID as parameters. It creates a new locomotive object in the locomotive database and sets the memory pointed to by pIDecoderObjectID to the decoder object ID used by the server as a key.  0KamDecoderPutDel  Parameter List Type Range Direction Description long 1 In Decoder object ID long 2 Out Pointer ID long 1 In Decoder Object ID long 2 Out Pointer ID long 2 Out Pointer ID long 1 In Decoder Object ID long 2 Out Pointer ID long 3 Out Pointer ID long 2 Out Pointer ID long 2 Out Pointer ID long 2 Out Pointer ID long 3 Out Pointer ID long 4 Out Pointer ID long 5 Out Pointer ID long 6 Out Pointer ID long 7 Out Pointer ID long 8 Out Pointer ID lo	
decoder model ID, and a pointer to a decoder object ID as parameters. It creates a new locomotive object in the locomotive database and sets the memory pointed to by pIDecoderObjectID to the decoder object ID used by the server as a key.  OKamDecoderPutDel Sarameter List Type Range Direction Description long 1 In Decoder object ID long 1 In Decoder ObjectID long 1 In Decoder ObjectID long 1 In Decoder ObjectID long 1 Out Pointer ObjectID long 1 O	
parameters. It creates a new locomotive object in the locomotive database and sets the memory pointed to by pilbecoderObjectID to the decoderObjectID of the decoderObjectID object ID used by the server as a key.    Washingtone   Value   V	50 is stored in the server. Commands such as KamEngGetSpeed
locomotive database and sets the memory pointed to by plDecoderObjectID to the decoder object ID used by the server as a key.  OKamDecoderPutDel	communicate only with the server, not the actual decoder.
plDecoderObjectID to the decoder object ID used by the server as a key.  0KamDecoderPutDel  Type Range Direction Description  long 1 In Decoder Object ID used by the engine using the KamCmdCommand. OKamEngGetSpeed  0KamEngGetSpeed  Type Range Direction Description  liDecoderObjectID long 1 In Decoder Object ID long In In In Decoder Object ID long In In In Decoder Object ID long In	
server as a key.  0KamEngGetSpeed  0KamEngGetSpeed  0KamEngGetSpeed  1 Type Range Direction Description  1 DecoderObjectID long 1 In Decoder object ID lopSpeed  1 DecoderObjectID long 1 Out Pointer  1 OkamEngGetSpeed  1 Type Range Direction Description  1 In DecoderObjectID long 1 Out Pointer  1 OkamEngGetSpeed  1 Type Range Direction Description  1 In DecoderObjectID long 1 Out Pointer  1 OkamEngGetSpeed  1 In DecoderObjectID long 1 Out Pointer  1 OkamEngGetSpeed  1 In Description Description  1 In DecoderObjectID long 1 Out Pointer  1 OkamEngGetSpeed  1 In DecoderObjectID long 1 Out Pointer  1 OkamEngGetSpeed  1 In DecoderObjectID long 1 Out Pointer  1 OkamEngGetSpeed  1 In DecoderObjectID long 1 Out Pointer  1 OkamEngGetSpeed  1 OkamEngGetSp	
OKamDecoderPutDel 5 Range Direction Description 1 DecoderObjectID 1 long 2 long 1 long	
Parameter List Type Range Direction Description IDecoderObjectID long 1 In Decoder ObjectID long 1 In In In Decoder ObjectID long 1 In	B
lDecoderObjectID long 1 In Decoder object ID lpSpeed int 2 Out Pointer	
iClearState int 2 In Clear state flag speed	speed
	•
	direction
2 0 - retain state, 1 - clear state. 60 1 Opaque object ID handle returned by	
Return Value Type Range Description 60 KamDecoderPutAdd.	KamDecoderrutAdd.
1 iError = 0 for success. Nonzero is an error number set to 14, 18, or 128 speed steps and matches the values	
(see KamMiscGetErrorMsg). defined by NMRA S9.2 and RP 9.2.1. 0 is stop and 1 is	
KamDecoderPutDel takes a decoder object ID and clear flag emergency stop for all modes.	
as parameters. It deletes the locomotive object specified  3 Forward is boolean TRUE and reverse is boolean	
by IDecoderObjectID from the locomotive database.  65 FALSE.  Particle Management of the Property of the Prope	
0KamDecoderGetMfgName Return Value Type Range Descri	Return Value Type Range Description

25

26

### -continued

#### APPLICATION PROGRAMMING INTERFACE Error flag iError short iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg). KamEngGetSpeed takes the decoder object ID and pointers to locations to store the locomotive speed and direction as parameters. It sets the memory pointed to by lpSpeed to the locomotive speed and the memory pointed to by lpDirection to the locomotive direction. 0KamEngPutSpeed Type Range Direction Description<sup>e</sup> Parameter List IDecoderObjectID long 1 Decoder object ID In Locomotive speed Locomotive direction int \* 3 iDirection In Opaque object ID handle returned by KamDecoderPutAdd. Speed range is dependent on whether the decoder is set to 14, 18, or 128 speed steps and matches the values defined by NMRA S9.2 and RP 9.2.1. 0 is stop and 1 is emergency stop for all modes. 3 Forward is boolean TRUE and reverse is boolean FALSE. Return Value Type Range Description Error flag iError short iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg). KamEngPutSpeed takes the decoder object ID, new locomotive speed, and new locomotive direction as parameters. It sets the locomotive database speed to Speed and the locomotive database direction to iDirection. Note: This command only changes the locomotive database. The data is not sent to the decoder until execution of the KamCmdCommand command. Speed is set to the maximum possible for the decoder if iSpeed exceeds the decoders range. 0KamEngGetSpeedSteps Parameter List IDecoderObjectID long int \* Type Range Direction Description Decoder object ID 14,28,128 Out of speed steps Opaque object ID handle returned by KamDecoderPutAdd. Return Value Туре Range Description Error flag iError short iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg). KamEngGetSpeedSteps takes the decoder object ID and a pointer to a location to store the number of speed steps as a parameter. It sets the memory pointed to by lpSpeedSteps to the number of speed steps 0KamEngPutSpeedSteps Parameter List Type Range Direction Description IDecoderObjectID long Decoder object ID iSpeedSteps int 14,28,128 In Locomotive speed steps Opaque object ID handle returned by KamDecoderPutAdd. Return Value Type Range Description Error flag iError short iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg). KamEngPutSpeedSteps takes the decoder object ID and a new number of speed steps as a parameter. It sets the number of speed steps in the locomotive database to iSpeedSteps Note: This command only changes the locomotive database. The data is not sent to the decoder until execution of the KamCmdCommand command. KamDecoderGetMaxSpeed returns the maximum possible speed for the decoder. An error is generated if an attempt is made to set the speed steps beyond this value. 0KamEngGetFunction Parameter List Type Range Direction Description Decoder object ID 1DecoderObjectID long iFunctionID int int \* Function ID number loFunction Out Pointer to function value 65

Opaque object ID handle returned by

KamDecoderPutAdd.

#### -continued

			-cont	inued				
	APPL	ICATION	N PROGR	AMMING I	NTERFACE			
5	2 FL is 0. F1-F8 are 1-8 respectively. Maximum for this decoder is given by KamEngGetFunctionMax. 3 Function active is boolean TRUE and inactive is boolean FALSE.							
	Return Value	Туре	Rai		Description			
10	iError short  1 iError = 0 f	1		or flag	, mmhar			
10	(see KamMiscGetE			o is an erro	numoci			
	KamEngGetFunction							
	ID, and a pointer to function state as pa							
	to by lpFunction to				·omios			
15	0KamEngPutFuncti Parameter List		D	Direction	Description			
	lDecoderObjectID	Type long	Range 1	In	Decoder object ID			
	iFunctionID	int	0-8 2	In	Function ID number			
	iFunction 1 Opaque obj	int ect ID ha	3 Indie retti	In rned by	Function value			
20	KamDecoderPutAd		Indic Tota					
20	2 FL 18 U. F 1-			tively. Maxi				
	this decoder is give 3 Function ac			UE and inac				
	boolean FALSE.							
	Return Value iError short	Type 1		nge or flag	Description•			
25	1 iError = 0 f	or succes	ss. Nonze	ro is an erro	r number			
	(see KamMiscGetE KamEngPutFunction			er object ID	a function			
	ID, and a new fund							
	specified locomotiv							
30	iFunction. Note: To locomotive database							
	until execution of		CmdCom	nand comma	and.			
	OKamEngGetFunct Parameter List	ionMax Type	Range	Direction	Description			
	lDecoderObjectID	long	1	In	Decoder object ID			
	piMaxFunction	int *	0–8	Out	Pointer to maximum function number			
35	1 Opaque obj	ect ID ha	andle retu	rned by				
	KamDecoderPutAc Return Value	ld. Type	D <sub>2</sub>	nge	Description			
	iError short	1		or flag	Description			
				ro is an erro	r number			
40	(see KamMiscGetl KamEngGetFuncti			oder object	ID and a			
	pointer to the max							
	sets the memory p maximum possible							
	decoder.			-				
45	OKamEngGetName Parameter List	е Туре	Range	Direction	Description			
	lDecoderObjectID	long	1	In	Decoder object ID			
	pbsEngName	BSTR *	• 2	Out	Pointer to locomotive name			
	1 Opaque obj		andle retu	rned by				
50	KamDecoderPutAc 2 Exact retur		enends on	language. It	is			
31	Cstring * for C++.				. 13			
	Return Value	Type		nge	Description			
	iError short  1 iError = 0	1 for succe		ror flag ro is an erro	r number			
	(see KamMiscGetl	ErrorMsg	g).					
5.	KamEngGetName the locomotive nar							
	pointed to by pbsl	EngName						
	OKamEngPutName Parameter List	е Туре	Range	Direction	Description•			
	lDecoderObjectID		1	In	Decoder object ID			
6	bsEngName 1 Opaque ob	BSTR	2 andle reti	Out irned by	Locomotive name			
	KamDecoderPutA		muoie iell	ance by				
			e depend	s on languag	ge. It is			
	LPCSTR for C++. Return Value	Туре	Ra	inge	Description			
6:	iError short	1	Er	ror flag	•			
0.	1 iError = 0	AUT SHCCE	as. INODZ6	io is an effe	n number			

iError = 0 for success. Nonzero is an error number

(see KamMiscGetErrorMsg).

27

-continued

long locomotive decoders. Return Value

(see KamMiscGetErrorMsg).

iError short

Type

alias address as parameters. It makes the decoder

Range

iError = 0 for success. Nonzero is an error number

KamEngPutConsistParent takes the parent object ID and an

Error flag

Description

28

#### APPLICATION PROGRAMMING INTERFACE KamEngPutName takes a decoder object ID and a BSTR as parameters. It sets the symbolic locomotive name to bsEngName 0KamEngGetFunctionName Parameter List Туре Range Direction Description 1DecoderObjectID Decoder object ID long In Function ID number In iFunctionID int BSTR \* 3 Out Pointer to pbsFcnNameString function name Opaque object ID handle returned by KamDecoderPutAdd. Campeconerrudu. 1 FL is 0, F1-F8 are 1-8 respectively. Maximum for this decoder is given by KamEngGetFunctionMax. 3 Exareturn type depends on language. It is Cstring \* for C++. Empty string on error. Return Value Type Range Description iError short Error flag iError\* = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg). KamEngGetFuncationName takes a decoder object ID, function ID, and a pointer to the function name as parameters. It sets the memory pointed to by pbsFcnNameString to the symbolic name of the specified function. 0KamEngPutFunctionName Parameter List 1ypIDecoderObjectID long int Type Range Direction Description Decoder object ID In 0-82 In Function ID number bsFcnNameString BSTR 3 In Function name 1 Opaque object ID handle returned by KamDecoderPutAdd. FL is 0. F1-F8 are 1-8 respectively. Maximum for this decoder is given by KamEngGetFunctionMax. Exact parameter type depends on language. It is LPCSTR for C++. Туре Range Description Return Value Error Flag iError short iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg). KamEngPutFunctionName takes a decoder object ID, function ID, and a BSTR as parameters. It sets the specified symbolic function name to bsFcnNameString. 0KamEngGetConsistMax Range Parameter List Type Direction Description IDecoderObjectID piMaxConsist long 1 int 2 Decoder object ID In Pointer to max consist number Opaque object ID handle returned by KamDecoderPutAdd. Command station dependent. Return Value Туре Description Range iError short Error flag iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg). KamEngGetConsistMax takes the decoder object ID and a pointer to a location to store the maximum consist as parameters. It sets the location pointed to by piMaxConsist to the maximum number of locomotives that can but placed in a command station controlled consist. Note that this command is designed for command station consisting. CV consisting is handled using the CV commands. 0KamEngPutConsistParent Range Parameter List Туре Direction Description lDCCParentObjID long 1 In Parent decoder iDCCAliasAddr 2 Alias decoder address Opaque object ID handle returned by KamDecoderPutAdd. 1-127 for short locomotive addresses. 1-10239 for

### -continued

	APPLIC	CATION	PROG	RAMMII	NG INTER	FACE				
. 5	specified by IDCCPa	rentOhi	ID the c	onsist pa	rent referre	ed				
	to by iDCCAliasAddr. Note that this command is designed									
	for command station consisting. CV consisting is handled									
	using the CV comma consist; the old pare					а				
	To delete a parent in									
10	consist, you must ad				te the old					
	parent using KamEn 0KamEngPutConsist		nsistRem	ioveObj.						
	Parameter List	Туре	Range	Direct	ion Desc	cription				
	lDCCParentObjID	long	1	In		nt decoder				
	IDCCOP(ID	long	1	In		ct ID oder object ID				
15	DCCObjID long 1 In Decoder object ID Opaque object ID handle returned by									
	KamDecoderPutAdd									
	Return Value 1 iError short 1	Гуре		inge ror flag	Desc	cription				
	1 iError = 0 fo	r succes			error num	рег				
20	(see KamMiscGetEr				avant ahias	+ ID				
	KamEngPutConsistC and decoder object I					t ID				
	decoder specified by	IDCCC	ObjID to	the cons	ist identific	ed				
	by IDCCParentObjII for command station									
	using the CV comm	ands. N	ote: This	comma	nd is inval	id if				
25	the parent has not be	een set								
	KamEngPutConsistF 0KamEngPutConsist		eOhi							
	Parameter List	Туре		Direct	ion Des	cription				
	lDecoderObjectID	long	1	In	Dec	oder object ID				
30	<ol> <li>Opaque obje KamDecoderPutAde</li> </ol>		indie ret	urnea by						
		Гуре	R	ange	Des	cription				
		l 		rror flag		ha-				
	1 iError = 0 fo (see KamMiscGetEr			cio is an	Citot num	bei				
	KamEngPutConsistI					t ID as				
35	a parameter. It remo lDecoderObjectID fi									
	command is designed					CV				
	consisting is handled the parent is remove					f				
	A. Commands t									
40					ands that					
	control accessory de things such as access									
	efficiency, a copy of	all the	engine '	variables		i				
	is stored in the serv				the come	r not				
	KamAccGetFunctio the actual decoder.									
45	the berver copy or t									
	all changes to the e- command.	ngine u	sing the	KamCmo	Command	Į.				
	0KamAccGetFuncti	on								
	Parameter List	Type				cription				
50	lDecoderObjectID iFunctionID	long int	1 0-31 2	In 2 In		oder object ID ction ID number				
	lpFunction	int *	3	Out	Poi	nter to function				
	1 Opaque obje	ct ID h	andle rei	need by	valt	ie				
	KamDecoderPutAde		unaio 10.	uined o,						
	2 Maximum fo		lecoder i	s given b	y					
55	KamAccGetFunctio 3 Function act		oolean T	RUE and	l inactive	is				
	boolean FALSE.									
		Type 1		ange rror flag	Des	cription				
	1 iError = 0 fo	or succe	ss. Nonz		error nun	iber				
60	(see KamMiscGetE	rrorMsg	;).							
	KamAccGetFunction ID, and a pointer to					псиоп				
	function state as pa	rameter	s. It sets	the mem	ory pointe	d				
	to by lpFunction to 0KamAccGetFuncti	the spe	cified fu	nction st	ate.					
			Range	Direc	tion Des	cription				

Туре Range

long int \*

IDecoderObjectID

pi Value

Direction

In

Out

Description

Decoder object ID

Function bit mask

29

30

iError = 0 for success. Nonzero is an error number

#### -continued -continued APPLICATION PROGRAMMING INTERFACE APPLICATION PROGRAMMING INTERFACE Opaque object ID handle returned by KamDecoderPutAdd KamDecoderPutAdd Exact return type depends on language. It is Cstring \* for C++. Empty string on error. Each bit represents a single function state. Туре Range Maximum for this decoder is given by Return Value Description KamAccGetFunctionMax. iError short 1 Error flag iError = 0 for success. Nonzero is an error number Return Value Туре Description Range iError short Error flag (see KamMiscGetErrorMsg). KamAccGetName takes a decoder object ID and a pointer to iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg). a string as parameters. It sets the memory pointed to by KamAccGetFunctionAll takes the decoder object ID and a pbsAccNameString to the name of the accessory. 0KamAccPutName pointer to a bit mask as parameters. It sets each bit in the memory pointed to by piValue to the corresponding Parameter List Турс Range Direction Description long BSTR function state lDecoderObjectID In Decoder object ID bsAccNameString 0KamAccPutFunction In Accessory name 1 Opaque object ID handle returned by KamDecoderPutAdd. Parameter List Type Range Direction Description 1DecoderObjectID Decoder object ID long iFunctionID int 0-31 2 In Function ID number Exact parameter type depends on language. It is iFunction Function value LPCSTR for C++. Opaque object ID handle returned by Return Value Туре Range Description KamDecoderPutAdd. iError short Error flag Maximum for this decoder is given by iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg). KamAccGetFunctionMax. Function active is boolean TRUE and inactive is KamAccPutName takes a decoder object ID and a BSTR as boolean FALSE. parameters. It sets the symbolic accessory name to bs AccName Return Value Туре Range Error flag 0KamAccGetFunctionName iError short iError = 0 for success. Nonzero is an error number Parameter List Range Direction Description Туре (see KamMiscGetErrorMsg). lDecoderObjectID long Decoder object ID KamAccPutFunction takes the decoder object ID, a function iFunctionID int 0 - 312In Function ID number BSTR \* 3 ID, and a new function state as parameters. It sets the pbsFcnNameString Pointer to specified accessory database function state to lFunction. function name 1 Opaque object ID handle returned by KamDecoderPutAdd. Note: This command only changes the accessory database. The data is not sent to the decoder until execution of the KamCmdCommand command. Maximum for this decoder is given by 0KamAccPutFunctionAll KamAccGetFunctionMax. Parameter List Type Range Direction Description Exact return type depends on language. It is IDecoderObjectID long Decoder object ID Cstring \* for C++. Empty string on error. Return Value Type Range Ĭ'n Pointer to function state Туре Description\* iError short Error flag array 1 Opaque object ID handle returned by KamDecoderPutAdd. iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg). KamAccGetFuncationName takes a decoder object ID, Each bit represents a single function state. function ID, and a pointer to a string as parameters. It sets the memory pointed to by pbsFcnNameString to the symbolic name of the specified function. Maximum for this decoder is given by Kam AccGet Function Max. Return Value Туре Description Range iError short Error flag 0KamAccPutFunctionName iError = 0 for success. Nonzero is an error number Parameter List Туре Range Direction Description (see KamMiscGetErrorMsg). lDecoderObjectID long Decoder object ID KamAccPutFunctionAll takes the decoder object ID and a iFunction ID int 0-31 2 In Function ID number BSTR bit mask as parameters. It sets all decoder function bsFcnNameString Function enable states to match the state bits in iValue. The 45 Opaque object ID handle returned by possible enable states are TRUE and FALSE. The data is not sent to the decoder until execution of the KamDecoderPutAdd. Maximum for this decoder is given by KamCmdCommand command KamAccGetFunctionMax. 3 Exact parameter type depends on language. It is LPCSTR for C++. 0KamAccGetFunctionMax Parameter List Direction Туре Range Description 1DecoderObjectID long Decoder object ID Return Value Туре Range Error flag piMaxFunction 0-31.2 Out Pointer to maximum iError short iError = 0 for success. Nonzero is an error number function number (see KamMiscGetErrorMsg). Opaque object ID handle returned by KamDecoderPutAdd. KamAccPutFunctionName takes a decoder object ID, function Maximum for this decoder is given by ID, and a BSTR as parameters. It sets the specified KamAccGetFunctionMax. symbolic function name to bsFcnNameString. Range Error flag 0KamAccRegFeedback Parameter List Type Return Value Type Description iError short Type Range Direction Description\* iError = 0 for success. Nonzero is an error number lDecoderObjectID long Decoder object ID In (see KamMiscGetErrorMsg). KamAccGetFunctionMax takes a decoder object ID and bsAccNode BSTR In Server node name iFunctionID 0-31 3 In Function ID number int pointer to the maximum function number as parameters. It Opaque object ID handle returned by sets the memory pointed to by piMaxFunction to the maximum possible function number for the specified KamDecoderPutAdd. Exact parameter type depends on language. It is LPCSTR for C++. decoder. 0KamAccGetName Maximum for this decoder is given by Parameter List Туре Range Direction Description KamAccGetFunctionMax. lDecoderObjectID long 1 BSTP \* 2 Decoder object ID Return Value Туре Description pbsAccNameString BSTP \* 2 Ou 1 Opaque object ID handle returned by Error flag Out Accessory name iError short

31

### 32

## -continued

### -continued

APPLICATION PROGRAMMING INTERFACE	APPLICATION PROGRAMMING INTERFACE			
(see KamMiscGetErrorMsg).	5 such as controlling command station power. The steps to			
KamAccRegFeedback takes a decoder object ID, node name	control a given command station vary depending on the			
string, and function ID, as parameters. It registers	type of command station.			
interest in the function given by iFunctionID by the method given by the node name string bsAccNode.	0KamOprPutTurnOnStation Parameter List Type Range Direction Descript	tion		
bsAccNode identifies the server application and method to	iLogicalPortID int 1-65535 1 In Logical			
call if the function changes state. Its format is	10 1 Maximum value for this server given by	•		
"\\{Server}\{App}.{Method}" where {Server} is the server	KamPortGetMaxLogPorts.	.•		
name, {App} is the application name, and {Method} is the	Return Value Type Range Descript	tion		
method name.	iError short 1 Error flag  1 iError = 0 for success. Nonzero is an error number			
0KamAccRegFeedbackAll Parameter List Type Range Direction Description	(see KamMiscGetErrorMsg).			
DecoderObjectID long 1 In Decoder object ID	15 KamOprPutTurnOnStation takes a logical port ID as a			
bsAccNode BSTR 2 In Server node name	parameter. It performs the steps necessary to turn on			
1 Opaque object ID handle returned by	the command station. This command performs a combinati	on		
KamDecoderPutAdd.	of other commands such as KamOprPutStartStation, KamOprPutClearStation, and KamOprPutPowerOn.			
2 Exact parameter type depends on language. It is LPCSTR for C++.	0KamOprPutStartStation			
Return Value Type Range Description	Parameter List Type Range Direction Descrip	tion		
iError short 1 Error flag	iLogicalPortID int 1-05535 I in Logical	port ID		
1 iError = 0 for success. Nonzero is an error number	1 Maximum value for this server given by			
(see KamMiscGetErrorMsg).  KamAccRegFeedbackAll takes a decoder object ID and node	KamPortGetMaxLogPorts.  Return Value Type Range Descrip	tion		
name string as parameters. It registers interest in all	iError short 1 Error flag			
functions by the method given by the node name string	1 iError = 0 for success. Nonzero is an error number			
bsAccNode bsAccNode identifies the server application	25 (see KamMiscGetErrorMsg).			
and method to call if the function changes state. Its format is "\\{Server}\{App\.{Method}\" where {Server} is	KamOprPutStartStation takes a logical port ID as a parameter. It performs the steps necessary to start the			
the server name, {App} is the application name, and	command station.			
{Method} is the method name.	0KamOprPutClearStation			
0KamAccDelFeedback	Parameter List Type Range Direction Descrip			
Parameter List Type Range Direction Description		port ID		
IDecoder Object ID long 1 In Decoder object ID bsAccNode BSTR 2 In Server node name	<ol> <li>Maximum value for this server given by KamPortGetMaxLogPorts.</li> </ol>			
iFunctionID int 0-31 3 In Function ID number	Return Value Type Range Descrip	tion		
1 Opaque object ID handle returned by	iError short 1 Error flag			
KamDecoderPutAdd.	iError = 0 for success. Nonzero is an error number			
2 Exact parameter type depends on language. It is	35 (see KamMiscGetErrorMsg). KamOprPutClearStation takes a logical port ID as a			
LPCSTR for C++.  3 Maximum for this decoder is given by	parameter. It performs the steps necessary to clear the			
KamAccGetFunctionMax.	command station queue.			
Return Value Type Range Description	0KamOprPutStopStation			
iError short 1 Error flag	Parameter List Type Range Direction Descrip			
1 iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg).	40 iLogicalPortID int 1-65535 1 In Logical  Maximum value for this server given by	port ID		
KamAccDeiFeedback takes a decoder object ID, node name	KamPortGetMaxLogPorts.			
string, and function ID, as parameters. It deletes	Return Value Type Range Descrip	tion		
interest in the function given by iFunctionID by the	iError short 1 Error flag			
method given by the node name string bsAccNode. bsAccNode identifies the server application and method to	1 iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg).			
call if the function changes state. Its format is	45 KamOprPutStopStation takes a logical port ID as a			
"\\{Server}\{App}.{Method}" where {Server} is the server	parameter. It performs the steps necessary to stop the			
name, {App} is the application name, and {Method} is the	command station.			
method name.	0KamOprPutPowerOn Parameter List Type Range Direction Descrip	tion		
0KamAccDelFeedbackAll Parameter List Type Range Direction Description•		port ID		
IDecoderObjectID long 1 In Decoder object ID	50 1 Maximum value for this server given by	•		
bsAccNode BSTR 2 In Server node name	KamPortGetMaxLogPorts.			
1 Opaque object ID handle returned by	Return Value Type Range Description in the Return of the Return Type Range Return Value Type Range Return Value Type Range Return Value Type Range Description in the Return Value Type Range Range Description in the Return Value Type Range R	tion		
KamDecoderPutAdd.  Exact parameter type depends on language. It is	1 iError = 0 for success. Nonzero is an error number			
LPCSTR for C++.	(see KamMiscGetErrorMsg).			
Return Value Type Range Description	55 KamOprPutPowerOn takes a logical port ID as a paramete	er.		
iError short 1 Error flag	It performs the steps necessary to apply power to the			
1 iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg).	track. 0KamOprPutPowerOff			
KamAccDelFeedbackAll takes a decoder object ID and node	Parameter List Type Range Direction Descrip	otion		
name string as parameters. It deletes interest in all	iLogicalPortID int 1-65535 1 In Logical	port ID		
functions by the method given by the node name string	Maximum value for this server given by			
bsAccNode, bsAccNode identifies the server application	KamPortGetMaxLogPorts.  Return Value Type Range Descrip	ntion		
and method to call if the function changes state. Its format is "\{Server}\{App}.{Method}" where {Server} is	iError short 1 Error flag			
the server name, {App} is the application name, and	iError = 0 for success. Nonzero is an error number			
{Method} is the method name.	(see KamMiscGetErrorMsg).			
A. Commands to control the command station	KamOprPutPowerOff takes a logical port ID as a paramet  65 It performs the steps necessary to remove power from the			
This section describes the commands that control the command station. These commands do things	track.			

33
-continued

34

-continued	-continued
ATION PROGRAMMING INTERFACE	APPLICATION PROGRAMMIN

			-						
APPLICATION	PROGRAMMING IN	TERFACE	<b>-</b> 5	AP	PLICATIO	N PROGI	RAMMING	INTERFACE	
0KamOprPutHardReset				6 - 16400 1				MARY	
71		Description		2 PARITYO -	-	- ODD, 2	- EVEN, 3	- MARK,	
iLogicalPortID int 1- 1 Maximum value tor t		ogical port ID		4 - SPACE 3 STOP 0		1.5 bits. 2	2 - 2 bits		
KamPortGetMaxLogPorts.	ins server given e,							ecommended	
Return Value Type	Range I	Description		value 2048					
iError short 1	Error flag		10				)FF, 2 - RT	S/CTS, 3 BOTH	
and the second s	s. Nonzero is an error n	umber		6 DATA 0 - 1			to	dahua 61a	
(see KamMiscGetErrorMsg).				7 DEBUGBi Bit 2 sends					
KamOprPutHardReset takes parameter. It performs the st		n a		queue data.				nows	
hard reset of the command s				reserved. B				al	
0KamOprPutEmergencyStop			15	sections. B					
Parameter List Type R		Description		8 shows co			O decimal is	5	
		ogical port ID		recommend 8 PARALLE		ugging.			
<ol> <li>Maximum value for t KamPortGetMaxLogPorts.</li> </ol>	inis server given by			8 PARALLE 0KamPortPutCo					
Return Value Type	Range I	Description		Parameter List		ange	Direction	Description•	
iError short 1	Error flag	ooo.ipo	•			-65535 1		Logical port ID	
	s. Nonzero is an error r	number	20	iIndex	int 2		In	Configuration type index	
(see KamMiscGetErrorMsg)				iValue	int 2		In	Configuration value	
KamOprPutEmergencyStop				iKey	int 3		In	Debug key	
parameter. It performs the st		ast					er given by		
an emergency stop comman	to all decoders.			KamPortGetMa  2 See FIG			uration Ind	ev values	
0KamOprGetStationStatus Parameter List Type	Range Direction	Description	25	for a table of in			manon inc	CA VAIDO	
iLogicalPortID int	1-65535 1 In	Logical port ID					ndex value.	Should be set	
pbsCmdStat BSTR *	2 Out	Command station		to 0.	,				
•		status string		Return Value	Туре	Ra	inge	Description	
1 Maximum value for	this server given by			iError short	1		ror flag	•	
KamPortGetMaxLogPorts.			20				ro is an em	or number	
	ends on language. It is		30	(see KamMiscG KamPortPutCor			ort ID. conf	ionration	
Cstring * for C++. Return Value Type	Range I	Description		index, configura					
iError short 1	Error flag	Description		sets the port pa					
	s. Nonzero is an error n	umber		specified by iValue. For the DEBUG iIndex value, the					
(see KamMiscGetErrorMsg)				debug file path	is C:\Temp	P			
KamOprGetStationStatus takes a logical port ID and a				is the physical		ID.			
pointer to a string as parameters. It set the memory				0KamPortGetC			<b>.</b>	m 1.1	
pointed to by pbsCmdStat to the command station status.  The exact format of the status BSTR is vendor dependent.				Parameter List		ange -65535 1	Direction	Description Logical port ID	
A. Commands to config		iLogicalPortID iIndex	int 2		In	Configuration type index			
communication port						configuration value			
	cribes the commands th	at	40				er given by		
configure the command stati	on communication port	. These	40	KamPortGetMa					
commands do things such a							uration Ind	ex values	
the commands in this section		oller		for a table of in				Description	
ID (iControllerID) to identif		Return Value iError short	Type i		ange ror flag	Description			
command station controller. The following table shows the mapping between the controller ID (iControllerID) and							ero is an er	ror number	
controller name (bsControllerName) for a given type of				5 (see KamMiscGetErrorMsg).					
command station controller.	, , ,			KamPortGetConfig takes a logical port ID, configuration					
		Description		index, and a po					
0 UNKNOWN	Unknown controller ty	pe		parameters. It s			ted to by pr	Value to	
1 SIMULAT 2 LENZ_1x	Interface simulator Lenz version 1 serial:	nunnari madula		the specified co 0KamPortGetN		value.			
2 LENZ_1x 3 LENZ_2x	Lenz version 2 serial		50	Parameter List		Range	Direction	n Description	
4 DIGIT_DT200	Digitrax direct drive s			iPhysicalPortID		1-65535	1 In	Physical port number	
5 DIGIT_DCS100	Digitrax direct drive s	upport using		pbsPortName	BSTR *	2	Out	Physical port name	
6 MASTERSERIES	DCS100 North coast engineering			1 Maximu KamPortGetMa			er given by	,	
	series	ig maser	55		turn type o	depends or	language.	It is	
7 SYSTEMONE 8 RAMFIX	System one RAMFIxx system			Return Value	++. Епріу Туре		ange	Description	
9 SERIAL	NMRA serial interface			iError short	1		ror flag		
10 EASYDCC	CVP Easy DCC							ror number	
11 MPK6050	Marklin 6050 interfac			(see KamMiscO					
12 MPK6023	Marklin 6023 interfac		60	KamPortGetNa					
13 DIGIT_PR1	Digitrax direct drive u			pointer to a por					
14 DIRECT Direct drive interface routine 15 ZTC ZTC system ltd				memory pointe name such as "			w ine pny	sicai port	
15 ZTC 16 TRIX	TRIX controller			OKamPortPutM					
	Value Values			Parameter List		Range	Directio	n Description	
0 RETRANS 10-255	<del>-</del>			iLogicalPortID	int	1-65535		Logical port ID	
1 RATE 0 - 300 BAUD,	1 - 1200 BAUD, 2 - 2		65		int	1-65535	2 In	Command station	
3 - 4800 BAUD, 4 - 9	600 BAUD, 5 - 14400 I	BAUD,						type ID	

35

36

		-continued		_			-cont	inued	
APP	LICATION P	ROGRAMMIN	G INTERFACE	_	AI	PLICATI	on progr	AMMING I	NTERFACE
iCommPortID  Maximum		535 3 In server given t	Physical comm port ID	5	0KamCmdCom Parameter List 1DecoderObject	Тур	e Range	Direction In	Description Decoder object ID
KamPortGetMaxl  See FIG. (	.ogPorts. 5: Controller I	D to controller	name		1 Opaque KamDecoderPu	object ID tAdd.	handle retur	•	
mapping for valugiven by KamMi	scMaxControl			10	Return Value iError short 1 iError =	Type  1  0 for succ	Ran Erro cess. Nonzer	or flag	Description
KamPortGetMax1	Physical.				(see KamMiscG	etErrorM	sg).		
Return Value iError short 1 iError = 0	Type 1 for success. 1	Range Error flag Vonzero is an e	Description rror number		KamCmdComm It sends all state the specified loo	e changes	from the ser	ver databas	
(see KamMiscGe KamPortPutMap(	tErrorMsg).			15	A. Cab Cor Thi	s section	mands describes con	mmands tha	t control
command station port ID as param	type ID, and	a physical com	munications		the cabs attache 0KamCabGetM		nmand statio	on.	
CommPortID for ControllerID.	the type of c				Parameter List iCabAddress	Type int	Range 1-65535 1		Cab address
)KamPortGetMa: Parameter List	LogPorts Type R	ange Directi	on Description•	20	pbsMsg 1 Maximu	BSTR *	2 s command :	Out station depe	Cab message string ndent.
piMaxLogicalPor		Out	Maximum logical port ID			turn type	depends on	language. It	
		turned on erro			Return Value	Туре	Rai	nge	Description
Return Value Error short	Type 1	Range Error flag	Description		iError short  1 iError =	1 0 for suc	Err cess. Nonzei	or flag o is an erro	r number
iError = 0	for success.	Nonzero is an	error number	25		etErrorM	sg).		
(see KamMiscGe KamPortGetMax		s a pointer to a	logical port		message string	as parame	ters. It sets	the memory	
D as a paramete iMaxLogicalPor					to by pbsMsg to 0KamCabPutM		ent cab mess	sage.	
KamPortGetMa	xPhysical				Parameter List	Турс	Range	Direction	
Parameter List MaxPhysical	Type R int * 1	ange Directi Out	on Description  Maximum physical	30	iCabAddress bsMsg	int BSTR	1 2	In Out	Cab address  Cab message string
•	int * 1	Out	port ID Maximum serial				s command: ype depends		
MaxSerial			port ID		LPCSTR for C	++.			
MaxParallel	int ** 1	Out	Maximum parallel port ID	35	Return Value iError short	Type 1		nge or flag	Description
l Normally Return Value		eturned on erro Range		33			cess. Nonze	ro is an erro	r number
Error short	Type 1	Error flag	•		KamCabPutMe	ssage take	s a cab addi		STR as
l iError = 0 (see KamMiscGe		Nonzero is an	error number		parameters. It s 0KamCabGetC		b message to	bsMsg.	
KamPortGetMax	Physical takes			40	Parameter List 1DecoderObject		Range 1	Direction In	Description* Decoder object ID
physical ports, th number of parall memory pointed	el ports as par	ameters. It sets	the		piCabAddress	int *	1-65535 2	2 Out	Pointer to Cab address
values A. Comman	ls that control	command flow	to the command		KamDecoderPu	ıtAdd.	handle retu	-	
station This	section descr	bes the comma	ands that	45	2 Maximu Return Value	ım value i Type	s command Ra	station depe nge	ndent. Descriptioni
control the comm	nand flow to t	he command s	ation. These		iError short	1		or flag	-
commands do th from the comma 0KamCmdConne	nd station.	onnecting and	disconnecting		1 iError = (see KamMisco KamCabGetCa	GetErrorM	lsg).		
Parameter List	Type Ran			50	to a cab addres	s as parar	neters. It set	the memor	y -
iLogicalPortID 1 Maximun KamPortGetMax	n value for thi	5535 1 In s server given	Logical port ID by	50	pointed to by p attached to the 0KamCabPutA	specified	decoder.	idless of the	cau
Return Value iError short	Type	Range Error flag	Description		Parameter List lDecoderObjec	Туре	Range 1	Direction In	Description Decoder object ID
1 iError = 0	) for success.	Nonzero is an	error number		iCabAddress	int	1-65535 2	2 In	Cab address
(see KamMiscGe KamCmdConnec	t takes a logic			55	KamDecoderPo	utAdd.	handle retu		
connects the ser- 0KamCmdDisCo		ified command	station.		2 Maximi Return Value	um value : Type	is command Ra	station depe	endent. Description
Parameter List	Type Ran		ion Description Logical port ID		iError short	1		ror flag	•
	n value for the	5535 1 In is server given		60	(see KamMisc	GetErrorN	ísg).		
KamPortGetMax Return Value	LogPorts. Type	Range	Description	30	KamCabPutAd address as para				
iError short	1	Error flag	-		by iDCCAddr	to the cab	specified by		
(see KamMiscG	etErrorMsg).	Nonzero is an			Th		describes m		commands
KamCmdDisCon It disconnects th				65	that do not fit 0KamMiscGet		ther categori	es.	
station.	- 5551 60 111	- PTTMICE COM	<del>_</del>		Parameter List		Range		Description

37

38

-continued

### -continued

-continued	_	-continued
APPLICATION PROGRAMMING INTERFACE	_	APPLICATION PROGRAMMING INTERFACE
Error int 0-65535 1 In Error flag 1 iError = 0 for success. Nonzero indicates an error.	5	automatically whenever the server stops running. Demo versions of the program cannot save data and this command
Return Value Type Range Description bsErrorString BSTR 1 Error string		will return an error in that case.  OKamMiscGetControllerName
Exact return type depends on language. It is		Parameter List Type Range Direction Description
Ostring for C++. Empty string on error.		iControllerID int 1-65535 1 In Command station
KamMiscGetErrorMsg takes an error flag as a parameter.	10	
t returns a BSTR containing the descriptive error		pbsName BSTR * 2 Out Command station typ
nessage associated with the specified error flag.		name
KamMiscGetClockTime		1 See FIG. 6: Controller ID to controller name
arameter List Type Range Direction Description LogicalPortID int 1-65535 1 In Logical port ID		mapping for values. Maximum value for this server is given by KamMiscMaxControllerID.
SelectTimeMode int 2 In Clock source	15	
iDay int * 0-6 Out Day of week	15	Cstring * for C++. Empty string on error.
iHours int * 0-23 Out Hours		Return Value Type Range Description
iMinutes int * 0-59 Out Minutes		pbsName BSTR 1 Command station type name
iRatio int * 3 Out Fast clock ratio		Return Value Type Range Description
Maximum value for this server given by [amPortGetMaxLogPorts.		iError short 1 Error flag
0 - Load from command station and sync server.	20	1 iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg).
- Load direct from server. 2 - Load from cached server		KamMiscGetControllerName takes a command station type ID
opy of command station time.		and a pointer to a type name string as parameters. It
Real time clock ratio.		sets the memory pointed to by pbsName to the command
eturn Value Type Range Description		station type name.
Error short 1 Error flag	25	0KamMiscGetControllerNameAtPort
iError = 0 for success. Nonzero is an error number see KamMiscGetErrorMsg).	20	Parameter List Type Range Direction Description iLogicalPortID int 1-65535 1 In Logical port ID
amMiscGetClockTime takes the port ID, the time mode, and		pbsName BSTR * 2 Out Command station ty
ointers to locations to store the day, hours, minutes,		name
nd fast clock ratio as parameters. It sets the memory		1 Maximum value for this server given by
pinted to by piDay to the fast clock day, sets pointed		KamPortGetMaxLogPorts.
by piHours to the fast clock hours, sets the memory	30	71
pinted to by piMinutes to the fast clock minutes, and		Cstring * for C++. Empty string on error.
the memory pointed to by piRatio to the fast clock ratio.  The servers local time will be returned if the command		Return Value Type Range Description iError short 1 Error flag
ation does not support a fast clock.		1 iError = 0 for success. Nonzero is an error number
KamMiscPutClockTime		(see KamMiscGetErrorMsg).
arameter List Type Range Direction Description	35	KamMiscGetControllerName takes a logical port ID and a
LogicalPortID int 1-65535 1 In Logical port ID		pointer to a command station type name as parameters. It
Day int 0-6 In Day of week Hours int 0-23 In Hours		sets the memory pointed to by pbsName to the command
Hours int 0-23 In Hours Minutes int 0-59 In Minutes		station type name for that logical port.  OKamMiscGetCommandStationValue
Ratio int 2 In Fast clock ratio		Parameter List Type Range Direction Description
Maximum value for this server given by	40	iControllerID int 1-65535 1 In Command station
KamPortGetMaxLogPorts. 2 Real time clock ratio.	40	type ID
teturn Value Type Range Description		iLogicalPortID int 1-65535 2 In Logical port ID
Error short 1 Error flag iError = 0 for success. Nonzero is an error number		iIndex int 3 In Command station array inde
ite KamMiscGetErrorMsg).		pi Value int * 0-65535 Out Command station value  1 See FIG. 6: Controller ID to controller name
amMiscPutClockTime takes the fast clock logical port,		mapping for values. Maximum value for this server is
e fast clock day, the fast clock hours, the fast clock	45	given by KamMiscMaxControllerID.
ninutes, and the fast clock ratio as parameters. It sets		2 Maximum value for this server given by
ne fast clock using specified parameters.		KamPortGetMaxLogPorts.
KamMiscGetInterfaceVersion arameter List Type Range Direction Description		3 0 to KamMiscGetCommandStationIndex .  Return Value Type Range Description
bsInterface Version BSTR * 1 Out Pointer to interface		Return Value Type Range Description iError short 1 Error flag
version string	50	
Exact return type depends on language. It is		(see KamMiscGetErrorMsg).
string * for C++. Empty string on error.		KamMiscGetCommandStationValue takes the controller ID,
eturn Value Type Range Description		logical port, value array index, and a pointer to the
iError short 1 Error flag iError = 0 for success. Nonzero is an error number		location to store the selected value. It sets the memory
ee KamMiscGetErrorMsg).		pointed to by piValue to the specified command station miscellaneous data value.
amMiscGetInterface Version takes a pointer to an	33	0KamMiscSetCommandStationValue
terface version string as a parameter. It sets the		Parameter List Type Range Direction Description
emory pointed to by pbsInterfaceVersion to the interface		iControllerID int 1-65535 1 In Command station
ersion string. The version string may contain multiple		type ID
nes depending on the number of interfaces supported.  KamMiscSaveData		iLogicalPortID int 1-65535 2 In Logical port ID iIndex int 3 In Command station array index
arameter List Type Range Direction Description	60	ilndex int 3 In Command station array index iValue int 0-65535 In Command station Value
ONE		1 See FIG. 6: Controller ID to controller name
eturn Value Type Range Description		mapping for values. Maximum value for this server is
Error short 1 Error flag		given by KamMiscMaxControllorID.
iError = 0 for success. Nonzero is an error number		2 Maximum value for this server given by
see KamMiscGetErrorMsg). amMiscSaveData takes no parameters. It saves all server	65	KamPortGetMaxLogPorts. 3 0 to
	33	
ata to permanent storage. This command is run		Return Value Type Range Description

39 -continued

#### 40

API	PLICATI	ON PROGR	AMMING	INTERFACE
iError short	1	Erro	or flag	
1 iError = 0	for suc	cess. Nonzer-	o is an erro	r number
(see KamMiscGe	tErrorM	sg).		
KamMiscSetCon	ımandSt	ation Value ta	kes the con	troller ID,
logical port, valu	e array i	index, and ne	w miscellar	neous
data value. It set	s the spe	cified comm	and station	data
to the value give	n by piV	/alue.		
0KamMiscGetCo	mmands	StationIndex		
Parameter List	Type	Range	Direction	
iControllerID	int	1-65535 1	ln	Command station
				type ID
iLogicalPortID	int	1-65535 2	In	Logical port ID
piIndex	int	0-65535	Out	Pointer to maximum index
1 See FIG.	6: Contr	oller ID to c	ontroller na	me
mapping for valu	ics. Max	imum value	for this serv	er is
given by KamMi	scMaxC	ontrollerID.		
		or this serve	given by	
KamPortGetMax	LogPorts	5.		
Return Value	Туре	Ran		Description
iError short	1	Error flag		
		cess. Nonzer-	o is an erro	r number
(see KamMiscGe				
KamMiscGetCon				
logical port, and				
maximum index.				
to the specified of	ommand	l station max	imum misce	elianeous

Parameter List Type Range Direction PrimaxControllerID int \* 1-65535 1 Out Direction Description Maximum controller type ID

See FIG. 6: Controller ID to controller name mapping for a list of controller ID values. 0 returned on error. Return Value

Range Type Description iError short Error flag

iError = 0 for success. Nonzero is an error number (see KamMiscGetErrorMsg). KamMiscMaxControllerID takes a pointer to the maximum

controller ID as a parameter. It sets the memory pointed to by piMaxControllerID to the maximum controller type 0KamMiscGetControllerFacility

Parameter List Type Range

data index.

0KamMiscMaxControllerID

iControllerID 1-65535 1 In Command station type ID pdwFacility long \* 2 Pointer to command station facility mask

Direction

Description

See FIG. 6: Controller ID to controller nam mapping for values. Maximum value for this server is given by KamMiscMaxControllerID.

- 0 CMDSDTA\_PRGMODE ADDR 1 CMDSDTA\_PRGMODE\_REG
- 2 CMDSDTA\_PRGMODE\_PAGE
- 3 CMDSDTA\_PRGMODE\_DIR 4 CMDSDTA\_PRGMODE\_FLYSHT
- 5 CMDSDTA\_PRGMODE\_FLYLNG
- 6 Reserved 7 - Reserved
- 8 Reserved
- 9 Reserved
- 10 CMDSDTA\_SUPPORT\_CONSIST
- 11 CMDSDTA\_SUPPORT\_LONG 12 CMDSDTA\_SUPPORT\_FEED
- 13 CMDSDTA\_SUPPORT\_2TRK
- 14 CMDSDTA\_PROGRAM\_TRACK 15 CMDSDTA\_PROGMAM\_POFF
- 16 CMDSDTA\_FEDMODE\_ADDR
- 17 CMDSDTA FEDMODE REG - CMDSDTA\_FEDMODE\_PAGE
- 19 CMDSDTA\_FEDMODE\_DIR 20 CMDSDTA\_FEDMODE\_FLYSHT
- 21 CMDSDTA\_FEDMODE\_FLYLNG
- 30 Reserved

31 - CMDSDTA\_SUPPORT\_FASTCLK

Return Value Туре Description

#### -continued

	APPLICATION PROGRAMMING INTERFACE						
5	iError short 1 Error flag						
	1 iError = 0 for success. Nonzero is an error number						
	(see KamMiscGetErrorMsg).						
	KamMiscGetControllerFacility takes the controller ID and						
	a pointer to the location to store the selected						
	controller facility mask. It sets the memory pointed to						
10	by pdwFacility to the specified command station facility mask.						

The terms and expressions which have been employed in the foregoing specification are used therein as terms of description and not of limitation, and there is no intention, in the use of such terms and expressions, of excluding equivalents of the features shown and described or portions thereof, it being recognized that the scope of the invention is defined and limited only by the claims which follow.

What is claimed is:

- 1. A method of operating a digitally controlled model railroad comprising the steps of:
  - (a) transmitting a first command from a first program to an interface:
- (b) transmitting a second command from a second program to said interface; and
- (c) sending third and fourth commands from said interface representative of said first and second commands, respectively, to a digital command station.
- 2. The method of claim 1, further comprising the steps of:
- (a) providing an acknowledgment to said first program in response to receiving said first command by said interface prior to sending said third command to said digital command station; and
- (b) providing an acknowledgment to said second program in response to receiving said second command by said interface prior to sending said fourth command to said digital command station.
- 3. The method of claim 2, further comprising the steps of:
- (a) selectively sending said third command to one of a plurality of digital command stations; and
- (b) selectively sending said fourth command to one of said plurality of digital command stations.
- 4. The method of claim 3, further comprising the step of 45 receiving command station responses representative of the state of said digitally controlled model railroad from said plurality of digital command stations.
  - 5. The method of claim 4, further comprising the step of comparing said command station responses to previous commands sent to at least one of said plurality of digital command stations to determine which of said previous commands it corresponds with.
    - 6. The method of claim 5, further comprising the steps of:
  - (a) maintaining a sending queue of commands to be transmitted to said plurality of digital command stations; and
  - (b) retransmitting at least one of said commands in said sending queue periodically until removed from said sending queue as a result of the comparison of said command station responses to previous commands.
- 7. The method of claim 6, further comprising the step of updating a database of the state of said digitally controlled model railroad based upon said receiving command station responses representative of said state of said digitally con-65 trolled model railroad.
  - 8. The method of claim 7, further comprising the step of providing said acknowledgment to said first program in

41

response to receiving said first command by said interface together with state information from said database related to said first command.

- 9. The method of claim 8 wherein said first command and said third command are the same command, and said second command and said fourth command are the same command.
- 10. A method of operating a digitally controlled model railroad comprising the steps of:
  - (a) transmitting a first command from a first program to an interface; and
  - (b) said interface selectively sending a second command representative of said first command to one of a plurality of digital command stations based upon information contained within at least one of said first and second commands.
- 11. The method of claim 10, further comprising the steps 15 of:
- (a) transmitting a third command from a second program to said interface; and
- (b) said interface selectively sending a fourth command representative of said third command to one of said plurality of digital command stations based upon information contained within at least one of said third and fourth commands.
- 12. The method of claim 10 wherein said first program and said interface are operating on the same computer.
- 13. The method of claim 11 wherein said first program, said second program, and said interface are all operating on different computers.
- 14. The method of claim 10, further comprising the step of providing an acknowledgment to said first program in response to receiving said first command by said interface prior to sending said second command to one of said plurality of said digital command stations.
- 15. The method of claim 10 wherein said interface communicates in an asynchronous manner with said first program while communicating in a synchronous manner with said plurality of digital command stations.
- 16. A method of operating a digitally controlled model railroad comprising the steps of:
  - (a) transmitting a first command from a first program to an 40 of interface;
  - (b) transmitting a second command from a second program to said interface; and
  - (c) said interface sending a third and fourth command representative of said first command and said second command, respectively, to the same digital command station.
- 17. The method of claim 16 wherein said interface communicates in an asynchronous manner with said first and second programs while communicating in a synchronous 50 manner with said digital command station.
- 18. The method of claim 16, further comprising the step of providing an acknowledgment to said first program in response to receiving said first command by said interface prior to sending said third command to said digital command 55 station.
- 19. A method of operating a digitally controlled model railroad comprising the steps of:
  - (a) transmitting a first command from a first program to a first processor; and
  - (b) said first processor providing an acknowledgment to said first program indicating that said first command has properly executed prior to execution of commands related to said first command by said digitally controlled model railroad.
- 20. The method of claim 19, further comprising the step of sending said first command to a second processor which

42 processes said first command into a state suitable for a digital command station.

- 21. The method of claim 19, further comprising the steps of:
  - (a) transmitting a second command from a second program to said first processor; and
  - (b) said first processor selectively providing an acknowledgment to said second program indicating that said second command has properly executed prior to execution of commands related to said second command by said digitally controlled model railroad.
- 22. The method of claim 21, further comprising the steps of:
  - (a) sending a third command representative of said first command to one of a plurality of digital command stations based upon information contained within at least one of said first and third commands; and
  - (b) sending a fourth command representative of said second command to one of said plurality of digital command stations based upon information contained within at least one of said second and fourth commands.
- 23. A method of operating a digitally controlled model railroad comprising the steps of:
  - (a) transmitting a first command from a first program to an asynchronous command processor;
- (b) said asynchronous command processor providing an acknowledgment to said first program indicating that said first command has properly executed prior to execution of said first command by said digitally controlled model railroad;
- (c) sending said first command to a command queue where said asynchronous command processor considers the intended destination device of said first command; and
- (d) processing said first command by said synchronous command processor into a suitable format for execution by a digital command station for said digitally controlled model railroad.
- 24. The method of claim 23 further comprising the steps
- (a) receiving responses from said digital command station; and
- (b) updating a first database of the state of said digitally controlled model railroad based upon said responses from said digital command station.
- 25. The method of claim 24, further comprising the steps of:
  - (a) sending a first response to said command queue from said synchronous command processor where said synchronous command processor considers said command queue the intended destination device of said first response; and
  - (b) processing said first response by said asynchronous command processor into a suitable format for said first program.
- 26. The method of claim 25, further comprising the step of updating a second database of the state of said digitally controlled model railroad by said asynchronous command processor based upon said first response from said synchronous command processor.
  - 27. The method of claim 26, further comprising the step of querying said second database by said asynchronous command processor providing said acknowledgment to said first program providing the information requested and not sending said first command to said command queue.

\* \* \* \* \*

### UNITED STATES PATENT AND TRADEMARK OFFICE **CERTIFICATE OF CORRECTION**

PATENT NO.

DATED

: 6,530,329 B2

: March 11, 2003

INVENTOR(S) : Katzer

Page 1 of 1

It is certified that error appears in the above-identified patent and that said Letters Patent is hereby corrected as shown below:

Column 2,

Line 38, change "model railroad In" to -- model railroad. In --

Signed and Sealed this

Fifteenth Day of March, 2005

JON W. DUDAS Director of the United States Patent and Trademark Office